



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



armasuisse

MASTER THESIS AT THE DEPARTMENT OF
INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING
COMPUTER ENGINEERING AND NETWORKS LABORATORY (TIK)
COMMUNICATION SYSTEMS GROUP (CSG)

MASTER THESIS

ON

“WIRELESS LINK QUALITY ESTIMATION IN MOBILE
NETWORKS”

PIRMIN HEINZER

Supervisor: Prof. Dr. Bernhard Plattner
Advisor: Dr. Vincent Lenders
Advisor: Dr. Franck Legendre
Time Period: November 2010 - Mai 2011

Abstract

Wireless Communication increasingly influences several aspects of our. As the frequency spectrum is limited for the popular technologies IEEE 802.15.4 and IEEE 802.11.x a huge effort is spent on routing, rate selection, handover, or jamming detection. Such algorithms heavily depend fast and accurate link quality estimation. In this thesis we show the strength and weaknesses of state of the art link quality estimation. By using the capabilities of software defined radio, we define a new link quality estimation approach based on chip errors per symbol (CEPS) for direct sequence spread spectrum transceivers represented by an IEEE 802.15.4 implementation. The new link quality estimator is evaluated on different link conditions, including multi-path, mobile and jamming scenarios. We show that the CEPS link quality estimator performs more accurate than received signal strength based estimators and much faster than the packet statistic based estimators by decreasing only slightly in accuracy.

Drahtlose Kommunikation nimmt mehr und mehr Einfluss auf verschiedene Aspekte unseres Lebens. Da das Frequenzspektrum für beliebte Technologien wie IEEE 802.15.4 und IEEE 802.11.x begrenzt sind wird viel Aufwand für Routing, Wahl der Übertragungsrate, Zugangspunktübergabe oder Störerkennung betrieben. Solche Algorithmen sind stark von schneller und präziser Einschätzung der Kanal Qualität (LQE) abhängig. Die vorliegende Arbeit zeigt die Stärken und Schwächen von LQE auf dem Stand der Technik. Wir verwenden software definierten Funk um einen neuen LQE Ansatz basierend auf Chip Fehlern pro Symbol (CEPS) für DSSS Empfänger zu definieren, welcher auf einer Implementation von IEEE 802.15.4 umgesetzt wird. Der neue LQE wird in verschiedenen Kanalbedingungen inklusive Mehrpfad-, beweglichen und Störszenarien evaluiert. Dabei zeigen wir, dass der CEPS LQE genauer als Schätzer basierend auf Signalstärke und schneller als Schätzer welche mit Packet Statistiken arbeiten, bei gleichzeitig nur geringer Einbusse an Genauigkeit.

Acknowledgment

I would like to thank my main advisor Dr. Vincent Lenders for the great support and for all the motivating discussion we had, Dr. Franck Legendre for the coordination at ETH Zürich and the helpful conversations. I would also like to express my gratitude Professor Dr. Bernhard Plattner for his support and all the contributing input.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 1 |
| 1.1 | Motivation | 1 |
| 1.2 | Goals | 2 |
| 1.3 | Thesis Structure | 3 |
| 2 | Link Quality Estimation | 5 |
| 2.1 | Estimators based on Signal Power | 6 |
| 2.1.1 | Metrics | 6 |
| 2.1.2 | Discussion | 8 |
| 2.2 | Estimators based on Packet Statistics | 8 |
| 2.2.1 | Metrics | 9 |
| 2.2.2 | Single Metric Estimators | 9 |
| 2.2.3 | Hybrid Estimators | 10 |
| 3 | IEEE 802.15.4 | 15 |
| 3.1 | IEEE 802.15.4 Standard | 15 |
| 3.2 | ZigBee | 17 |
| 4 | GNU Radio IEEE 802.15.4 SDR | 19 |
| 4.1 | Software Defined Radio | 19 |
| 4.2 | IEEE 802.15.4 Implementation on GNU Radio USRP2 Platform | 20 |
| 4.2.1 | GNU Radio | 20 |
| 4.2.2 | USRP2 | 21 |
| 4.2.3 | UCLA ZigBee Project | 22 |
| 5 | Exploration of SDR Capabilities for Link Quality Estimation | 25 |
| 5.1 | Experimental Scenarios | 25 |

| | | |
|----------|---|-----------|
| 5.1.1 | Attenuated Cable Connection | 26 |
| 5.1.2 | Indoor Line of Sight | 26 |
| 5.1.3 | Indoor None Line of Sight | 27 |
| 5.1.4 | Outdoor | 27 |
| 5.1.5 | Outdoor Mobile | 27 |
| 5.2 | Chip Errors | 28 |
| 5.2.1 | Distribution of Chip Errors | 28 |
| 5.2.2 | Burstiness of Chip Errors | 30 |
| 5.3 | Symbol Errors | 33 |
| 5.3.1 | Distribution of Symbol Errors | 33 |
| 5.3.2 | Burstiness of Symbol Errors | 33 |
| 5.4 | Digital SNR | 34 |
| 5.5 | Synchronization Errors | 35 |
| 5.6 | LQE Potential of SDR provided Information | 38 |
| 6 | New SDR Defined LQE Models | 39 |
| 6.1 | Metric Definition | 39 |
| 6.1.1 | Chip Errors per Symbol (CEPS) | 39 |
| 6.1.2 | Digital SNR | 40 |
| 6.2 | Modeling Link Quality Estimators | 40 |
| 6.2.1 | CEPS based Models | 40 |
| 6.2.2 | Digital SNR based Models | 44 |
| 6.2.3 | Combined Metric Models | 46 |
| 7 | Comparison to Standard LQE | 49 |
| 7.1 | Choosing Competitive Standard LQE's | 49 |
| 7.2 | Implementing Standard Link Quality Estimators | 50 |
| 7.2.1 | WMEWMA | 50 |
| 7.2.2 | ETX | 50 |
| 7.2.3 | RNP | 51 |
| 7.2.4 | Four-Bit | 53 |
| 7.3 | Performance Comparison of Link Quality Estimators | 54 |
| 7.3.1 | Scenario Analysis | 54 |
| 7.3.2 | Discussion of Precision and Timescale Performance | 58 |

| | | |
|----------|--|-----------|
| 7.4 | Network Performance Measurement Experiment | 58 |
| 7.4.1 | Integration of CEPS LQE into GNU Radio | 59 |
| 7.4.2 | Experimental Setup | 59 |
| 7.4.3 | Performance Evaluation | 60 |
| 8 | Conclusion and Outlook | 63 |
| 8.1 | Conclusion | 63 |
| 8.2 | Outlook | 64 |
| A | CD-ROM Content | 65 |
| B | Abbreviations | 67 |

List of Figures

| | | |
|------|--|----|
| 2.1 | Signal strength values in different directions [13] | 7 |
| 2.2 | FLQE Membership Functions [29] | 11 |
| 2.3 | Geometric Combination of SNR and LQI of the Triangle Metric | 12 |
| 3.1 | IEEE 802.15.4 symbol to chipsequence conversion | 16 |
| 3.2 | Frame Layout of an IEEE 802.15.4 Packet | 17 |
| 4.1 | Block diagram of a USRP2 | 21 |
| 4.2 | GnuRadio IEEE 802.15.4 Demodulation [36] | 22 |
| 4.3 | Block schema of of the modulator implemented in GNU Radio [36] | 23 |
| 5.1 | Setup cable connection scenario | 26 |
| 5.2 | indoor office scenarios | 27 |
| 5.3 | Outdoor measurement scenario | 27 |
| 5.4 | mobile measurement scenario | 28 |
| 5.5 | Chip error distribution in a payload symbol | 29 |
| 5.6 | Chip Error Distribution in one Packet | 29 |
| 5.7 | Chip Error per Packet in a Measurement Series | 30 |
| 5.8 | Chip Errors per Symbol in Payload Distribution | 31 |
| 5.9 | Chip Error distribution of 9 consecutive Symbols | 31 |
| 5.10 | Conditional Probability distribution of received chips | 32 |
| 5.11 | Chip Beta factor distribution | 32 |
| 5.12 | Symbol Errors during a Measurement Run for a 10% PRR channel | 33 |
| 5.13 | Comparison of beta-factors for different Link Qualities | 34 |
| 5.14 | SNR behavior during one measurement run | 35 |
| 5.15 | Digital SNR Distribution | 36 |
| 5.16 | Chip Errors per Frame during Synchronization Phase | 37 |

| | | |
|------|---|----|
| 5.17 | Number of Synchronization Zeros detected | 37 |
| 6.1 | Linear regression plots of cable and line of sight measurements | 41 |
| 6.2 | Precision of Estimation for linear regression and simple geometric models | 42 |
| 6.3 | Comparison of worst symbol model performance | 43 |
| 6.4 | Number of Symbols Precision Trade off | 44 |
| 6.5 | Comparison of estimation precision of the dSNR and CEPS models | 45 |
| 6.6 | Logistic model through curve fitting | 46 |
| 6.7 | Correlation and precision of SDRLQE | 47 |
| 6.8 | Combination of dSNR and CEPS by division | 48 |
| 7.1 | Correlation of the WMEWMA values to the actual PRR | 51 |
| 7.2 | Correlation of ETX values to the actual PRR | 52 |
| 7.3 | Correlation of RNP values to the actual PRR | 52 |
| 7.4 | Correlation of Four-Bit values to the actual PRR | 53 |
| 7.5 | Error distribution of different LQE on a cable connection | 55 |
| 7.6 | Error distribution of different LQE on a wireless LOS connection | 56 |
| 7.7 | Error distribution of different LQE on a wireless NLOS connection | 56 |
| 7.8 | Error distribution of different LQE on an outdoor wireless LOS connection | 57 |
| 7.9 | Error distribution of different LQE on a mobile wireless connection | 58 |
| 7.10 | Iperf experimental setup | 59 |
| 7.11 | Estimator values in jammed timeseries | 61 |
| 7.12 | estimation precision on static and jammed iperf measurements | 61 |
| 7.13 | estimation precision in long term jammed iperf measurement | 62 |

1

Introduction

1.1 Motivation

Smartphone statistics of IDC's Worldwide Quarterly Mobile Phone Tracker for Q4 2010 [1] state that for the first time ever smartphone sales exceeded the PC sales. At the same time Wireless Sensor Networks (WSN) become more and more omnipresent in area monitoring or in machine health monitoring in industry.

Such a development brings us closer to the internet of things and ubiquitous computing, but at the same time, the frequency spectrum is limited and therefore its efficient use and distribution is paramount. In the recent past, wireless technologies like IEEE 802.15.4 for sensors or IEEE 802.11x in WLAN networks have gained tremendous popularity. To coordinate all the communicating devices we try to improve the performance of routing, rate selection, handover or jamming detection algorithms. For the algorithms to make the right decisions, they heavily depend on link quality estimators. Link quality estimation is a fundamental problem of such algorithms and common available implementations often do not suite the given application and given environmental conditions. However, estimating the effective link quality in real-life wireless networks is quite a challenging tasks given the unpredictable and location-sensitive nature of wireless channels. Instability of links and connectivity in low power WSNs has so far been regarded as a difficult problem that existing routing algorithms try their utmost to avoid. Therefore, research has mainly focused on link quality estimation and routing techniques [2–5], which identify and utilize consistently high quality links for packet forwarding. This only makes sense when we assume a static setup, without moving nodes and changes in the environment. Links of intermediate quality (packet reception rate between 10% and 90%) are ignored to ensure routing stability and to attain high end to end reliability.

Studies like [6] have shown that these intermediate quality links are bursty i.e., they frequently switch between stable and unstable periods of transmission for a limited amount of time or a small number of consecutive packets. Existing techniques that rely on the received signal strength [7] are relatively fast but not accurate in mobile environments with signal fading and interference. Active packet probing techniques are better with this regard but require much more time and produce more overhead, which is not suitable when we imagine a handover scenario of a mobile node.

In order to fully exploit the fact, that modern sensor carrying devices like smartphones have several different radio chips installed. Another part of research devoted itself to the field of cognitive radios (CR) [8]. It is useful to visualize a CR as a software defined radio (SDR) operating under the control of an intelligent software packet called a cognitive engine. The cognitive engine includes a set of algorithms that perform the sensing, learning, optimization and adaptation control of the CR. You can think of the cognitive engine as continually adjusting the parameters of the SDR, observing the results and taking actions to move the radio toward some desired operational state while learning in the process. We will partly adapt this concept for link quality estimation by adjusting parameters and observing results of a software defined radio to find new ways of describing link quality.

1.2 Goals

The goal of this thesis is to develop a new wireless link quality estimator that manages to quickly and accurately estimate the achievable link performance in mobile networking environments. In particular we are interested in estimating the probability that a packet may successfully be received at the receiver. To achieve a rapid estimation, we explore the feasibility to model this probability using measurements from the chip errors at the physical layer. Chips are the smallest unit of transmitted data and represent bits of information in a direct sequence spread spectrum. We will explore IEEE 802.15.4 communication where four bits (one symbol) are represented as a sequence of 32 chips. Since many more chips are sent than packets per time unit, we expect a huge increase in time for the estimation of the probability of packet errors due to the much larger sampling size of the wireless channel. Ideally, we will be able to estimate accurately the probability of successful delivery based on just a few bits at the packet's physical layer preamble. A general challenge in estimating the probability of successful packet delivery in wireless networks is to find a model that is accurate across the full range of delivery probabilities between 0% and 100 % PRR in different environments. Therefore, we take special care in developing a model that produces a more accurate and less ambiguous estimation of the PRR. We will define the estimator in a fashion that it is robust to different environment including multi-path fading, environments with node mobility, and networks with possible interference/jamming. The model will be empirically validated for different scenarios.

1.3 Thesis Structure

The remainder of this thesis is compiled the following way. Chapter 2 gives an overview of currently used link quality metrics and estimators. Points out their strengths and weaknesses. In Chapter 3 we present the characteristics of the IEEE 802.15.4 communication standard in service for example in ZigBee applications. Chapter 4 we introduce the concept of software defined radios the used GNU Radio USRP2 platform to conduct our experiments and the UCLA ZigBee Project implementing IEEE 802.15.4 on this platform. We use this setup in Chapter 5 to explore the capabilities such a software defined radio provides to link quality estimation. Based on these observations, we define new metrics and models to express link quality in Chapter 6. In Chapter 7 we compare our new model to established link quality estimators. All findings will be concluded and an outlook of future work left to do will be provided in the final chapter.

2

Link Quality Estimation

In wireless communication, interference, multi-path effects, harsh environment conditions have a huge impact on the correct signal propagation. Interference is caused when simultaneous transmissions are done by near operating wireless networks sharing the same transmission channel. Multi-path effects consist in signals absorption, attenuation, reflection and scattering when hurting obstacles, smooth and rough surfaces and were often found where wireless sensors are distributed. The more power we apply to signal, the more resilient it will be against these communication hurdles. However, in Wireless Sensor Networks and because of stringent cost and energy constraints, sensor nodes are limited by a low-power radio transceiver which emits very low power signals. Therefore, while propagating through the wireless medium, these signals are too weak to be uphold against unwanted modifications. Consequently and comparing to other wireless networks, the quality of radio links found in wireless sensor networks experiences erratic variation over time and space making the communication over them very unreliable. Different goals for optimization on several platforms and application lead to numerous studies [9–19] increasing the knowledge about radio link unreliabilities by trying to grasp their particular characteristics. We evaluate the results of this related work on link quality estimation. We divide it by first presenting characteristics, metrics and estimators based on Signal Power measurement implemented most of the time in hardware directly on a radio chip. In the following section we analyze the gained insights of studies concentrating on different interpretations of packet statistics detected in some networks. In this section we show as well higher layer estimators which combine several information building hybrid link quality estimators.

2.1 Estimators based on Signal Power

Almost every theoretical wireless channel model is based on the SNR. Recent sophisticated studies [20] present for example an analytical formula for predicting Packet Error rate in Wireless networks where convolutional codes are used jointly with Viterbi decoder over an AWGN channel. Their approach was based on a precise analysis of the error process at the output of the Viterbi decoder. This formula was shown to precisely predict the PER as a function of convolutional code parameter and SNR over the AWGN channel. The authors expand their model to the case of fading channel under block fading hypothesis in [21]. Although these models do not cover the whole scope of a hardware realization, a logical first step in Link quality estimation is measuring the signal strength of a transmission as a basic indicator. In modern radio chips as the CC2420 [22] signal power is represented by the Received Signal Strength Indicator (RSSI). The characteristics of such metrics involve problems to the SNR based theoretical models. Such characteristics include

- **Non-Isotropic Connectivity:** The Received Signal Strength Indicator varies continuously when incrementally changing of the propagation direction from the sender [13]. This irregularity is presented in Figure 2.1 This variability leads to non isotropic packet reception ratio (PRR) which means that the PRR does not have the same value in all directions from the source.
- **Transitional Region:** Wireless sensor network protocols are often evaluated through simulations that make simplifying assumptions about the link layer. Several empirical studies [4, 12, 23] have questioned the validity of these assumptions. These studies have revealed the existence of three distinct reception regions in a wireless link: connected, transitional, and disconnected. The transitional region is often quite significant in size, and is generally characterized by high-variance in reception rates and asymmetric connectivity. Particularly, in dense deployments such as those envisioned for sensor networks, a large number of the links in the network (even higher than 50%) can be unreliable due to the transitional region.

2.1.1 Metrics

The following metrics based on signal power are often used to assess link quality.

- **RSSI:** Acronym of the Received Signal Strength Indicator, RSSI is read from an 8 bit register in the radio chip. For Zigbee Application the CC2420 [22] radio chip from chipcon is often integrated in the sensor. Its RSSI Acquisition starts by sampling the signal strength over the first 8 symbols following the start frame delimiter (SFD) which marks the end of the synchronization symbols of the received packet. The measured values are averaged and stored in reserved register *RSSI_{VAL}*. RSSI can provide a quick estimate of whether a downlink is in the connected region or not, as shown for example in [24].

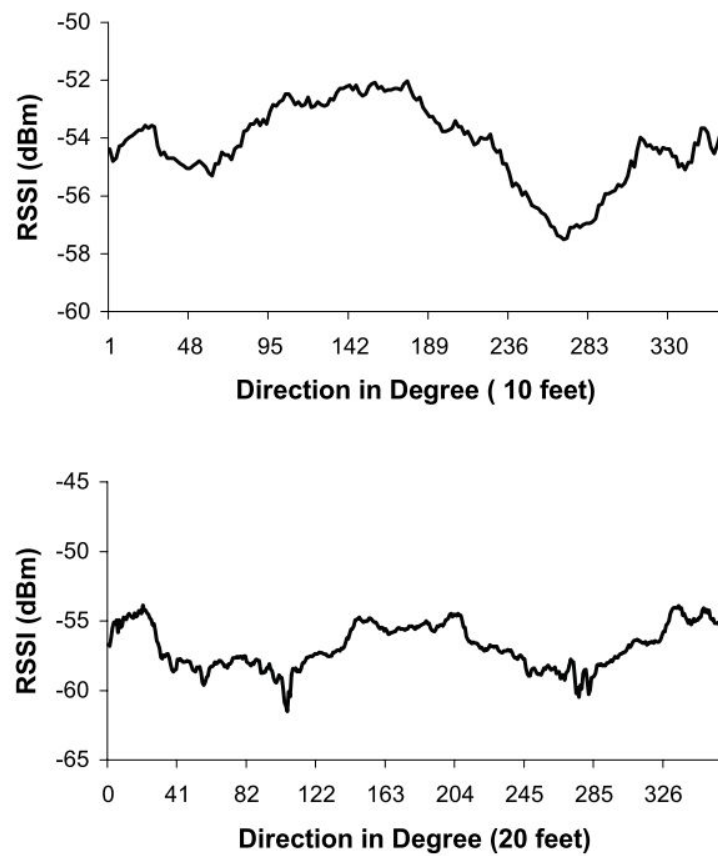


Figure 2.1: Signal strength values in different directions [13]

- **LQI:** Abbreviation of Link Quality Indicator. LQI is not directly dependent on signal power but it is provided by hardware as the CC2420 as well. CC2420 LQI indicates the average correlation value of the 8 first symbols of the received PHY header. The data sheet does not mention the exact algorithm. Some researchers argue that the LQI is more relative to PRR than RSSI [25]. But most of these studies give different conclusions. The correlation coefficient between LQI to PRR varies a lot in these studies [15, 25]. Hence, the trustworthiness of the LQI is at least unclear.
- **SNR:** Signal to Noise Ratio or sometimes SIR for Signal to Interference Ratio is the most used link quality metric in theoretical channel modeling. In practice it is measured using oscilloscopes or for the case of hardware providing RSSI values, one measures the RSSI between packet transmissions to provide a value for the noise and therefore being able to compute the current SNR as extensively described in [26].

2.1.2 Discussion

Such mostly hardware generated link quality metrics allow a quick estimation of link quality, but is in practice limited to differentiate between a high quality link and everything else. So the interesting links in the transitional region are left out. There is a general belief in the Wireless Sensor Network community that the RSSI is a bad estimator of the link quality. This belief is due to the existence of many asymmetry links in older radios such as CC1000 and TR1000. Newer radios are based on IEEE 802.15.4 standard such as CC2420 implement another parameter called LQI which is believed to be a better indicator than RSSI. Srninivasan et al. has conducted an evaluation of these claims in [7]. Their preliminary results indicate that RSSI for a given link has very small variation over time for a link. The results also indicate that when the RSSI is above the the sensitivity threshold (in their case -87 dBm) the packet reception rate (PRR) is at least 85%. Around this sensitivity threshold, however, the PRR is not correlated possibly due to variation in local phenomena such as noise. LQI, on the other hand varies over a wider range over time for a given link. However, the mean LQI computed over many packets has a better correlation with PRR.

2.2 Estimators based on Packet Statistics

Due to the unreliability of signal power based link quality estimation, a lot of higher layer application rely on metrics based on packet statistics. We introduce some of these metrics and present different realizations of existing estimators either based on single metric or hybrid estimators, using several metrics and combining knowledge of the physical, MAC and transport layer.

2.2.1 Metrics

Packet statistic based metrics all count in some way how many packets of the totally transmitted ones are received successfully. It is called packet success rate (PSR), packet delivery ratio (PDR), packet error rate (PER) or packet reception rate (PRR). We will use the PRR to describe the basic packet statistic in the reminder of this thesis and as main expression of end to end link quality as the more packets are successfully received, the more information is exchanged at the end of the day.

- **PRR:** Packet Reception Ratio (PRR) is defined as the number of successfully received packets over the number of sent packets . the number of sent packets is the sum of received and lost packets. The receiver infers the losses in packet reception by tracking the sequence numbers and counting gaps between them. PRR estimates are computed for each window of w received packets, as follows:

$$PRR(w) = \frac{\text{Number of received packets}}{\text{Number of sent packets}}$$

2.2.2 Single Metric Estimators

Simple link quality estimators rely only on a single metric similar to a modified PRR, three well known examples are WMEWMA, ETX and RNP presented below.

1. **WMEWMA [10]:** Acronym of Window Mean Exponential Weighted Moving Average, WMEWMA uses a Exponential Weighted Moving Average (EWMA) filter to combine recently and previously PRR computed estimates. We say that WMEWMA smooths PRR estimates in order to provide a transient fluctuations resistant metric. The EWMA filter is a function that gives greater or lower weight to more recent measurements and exponentially decreases respectively increases the weight of older ones. The rate of the decrease is governed by a smoothing factor α which ranges between 0 and 1. WMEWMA is calculated as follows:

$$WMEWMA = \alpha \times WMEWMA + (1 - \alpha) \times PRR.$$

2. **ETX [27]:** the Expected Transmission Count (ETX) approximates the required number of retransmissions to successfully deliver a packet. ETX takes into account link asymmetry by estimating the uplink quality from the sender to the receiver, denoted as PRR_{up} , as well as the downlink quality from the receiver to the sender, denoted as PRR_{down} . The values are generated by flooding estimation beacons throughout the network. By combining PRR_{up} and PRR_{down} , ETX provides an estimation of the bidirectional link quality, expressed as follows:

$$ETX = \frac{1}{PRR_{down} \times PRR_{up}}$$

3. **RNP [28]:** RNP counts the required number of packet transmissions and re-transmissions before a successful reception. This metric is computed for each w transmitted and retransmitted packets defined as follows.

$$RNP(w) = \frac{\text{Number of transmitted and retransmitted packets}}{\text{number of successfully received packets}} - 1$$

The number of successfully sent packets is determined by the sender as the number of acknowledged packets.

2.2.3 Hybrid Estimators

Hybrid estimators combine several metrics to cope with weaknesses of a single metric. They use for example sent and received traffic, combine physical and higher layer information to evaluate link quality. We present three examples the Fourbit Estimator, the Fuzzy Link Quality Estimator and the Triangle metric.

1. **Fourbit [2]:** Fourbit provides information, as the name implies in the form of four bits 1 from the physical, 1 from the link layer and 2 from the network layer. The physical layer can provide a rough measure of whether a link might be of high quality, enabling a link estimator to avoid spending effort on marginal or bad links. Once the estimator has gauged the quality of a link, the network layer can in turn decide which links are valuable for routing and which are not. The Fourbit estimator is an estimator computed at the sender node and approximates the packet retransmissions count based on statistics collected from received and sent packets. In fact based on w_{down} received packets, the node computes the WMEWMA estimate and derives an approximation of the RNP denoted as $estETX_{down}$.

$$estETX_{down} = \frac{1}{WMEWMA} - 1$$

Further, the sender computes RNP, denoted as $estETX_{up}$, based on w_{up} transmitted and retransmitted data packets to the receiver. Finally, Fourbit combines both $estETX_{up}$ and $estETX_{down}$ metrics via the EWMA filter in order to obtain an estimate of the bidirectional link expressed as follows:

$$FourBit(a, w) = \alpha \times FourBit + (1 - \alpha) \times estETX$$

Where $estETX$ corresponds to $estETX_{up}$ or $estETX_{down}$ accordingly.

2. **FLQE [29]:** The Fuzzy Link Quality Estimator, a recently proposed estimator, combines four link quality properties namely packet delivery (SPRR), asymmetry level (ASL), stability factor (SF), and channel quality quantified by SNR (ASNR). SPRR defines the capacity of the link to successfully deliver data, ASL represents the difference in connectivity between the uplink and the downlink, SF quantifies the variability level of of the link and the ASNR reflects the degree of noise in the

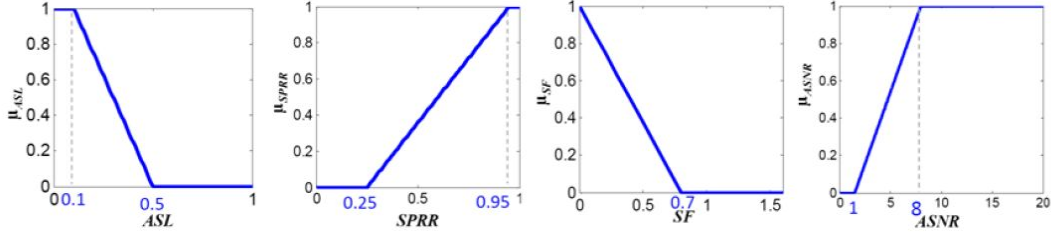


Figure 2.2: FLQE Membership Functions [29]

communication channel. Each of these properties are defined as a fuzzy variable. The overall quality of the link is traduced in a fuzzy rule:

IF the link has high packet delivery AND low asymmetry AND high stability AND high channel quality THEN it has high quality

The above rule is translated to the equation below:

$$\begin{aligned} \mu(i) &= \beta * \min(\mu_{SPRR}(i), \mu_{ASL}(i), \mu_{SF}(i), \mu_{ASNR}(i)) \\ &+ (1 - \beta) * \text{mean}(\mu_{SPRR}(i), \mu_{ASL}(i), \mu_{SF}(i), \mu_{ASNR}(i)) \end{aligned}$$

where: $\mu_{SPRR}(i)$, $\mu_{ASL}(i)$, $\mu_{SF}(i)$ and $\mu_{ASNR}(i)$ are the membership functions of the fuzzy variables. the definition of these functions is displayed in Figure 2.2.

Finally, FLQE uses a EWMA filter to smooth its estimates as follows:

$$FLQE(w) = \alpha * FLQE + (1 - \alpha) * 100 * \mu(i)$$

3. **Triangle Metric [30]:** The Triangle metric is a metric that combines geometrically the information of PRR, LQI and SNR into a robust estimator that guarantees fast and reliable assessment of link quality. The geometric combination is presented in Figure 2.3. The formal description of the triangle metric is the following: let us assume that n packets are used to sample the channel and m of those packets are successfully received ($0 < m \leq n$). The LQI and SNR of each successfully received packet i are denoted by lqi_i and snr_i . Upon reception of the sampling packets the receiver calculates the window mean SNR and LQI in the following way:

$$\overline{LQI}_w = \frac{\sum_{k=1}^m lqi_k}{n} \quad (2.1)$$

$$\overline{SNR}_w = \frac{\sum_{k=1}^m snr_k}{n} \quad (2.2)$$

Then, the receiver calculates the distance to the origin (length of the hypotenuse):

$$d_{\Delta} = \sqrt{\overline{SNR}_w^2 + \overline{LQI}_w^2}$$

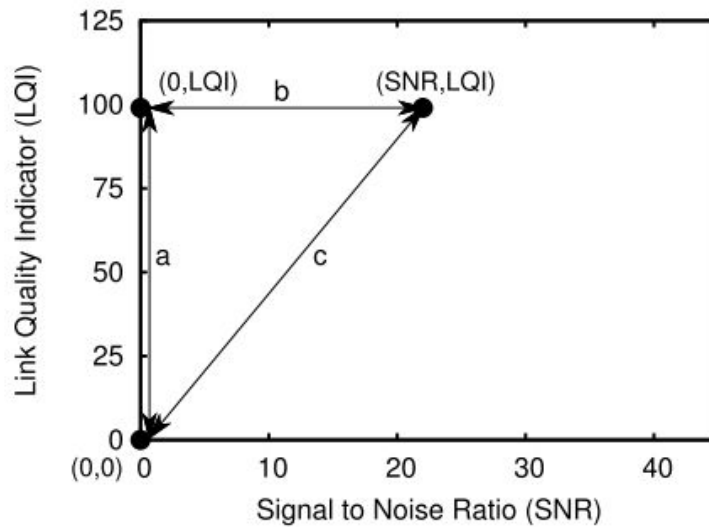


Figure 2.3: Geometric Combination of SNR and LQI of the Triangle Metric

The result of studies conducted by the inventors show that the triangle metric provides a quick and reliable estimation with as few as 10 packets and that it performs well both in static and dynamic environments.

In comparative simulation study of link quality estimators [31], it has been argued that although RSSI, LQI and SNR have the advantage of not requiring additional computational overhead, they are judged as not sufficient to characterize the holistic link quality link quality as they are measured uniquely based on the first 8 symbols of a received packet and not the whole packet. Packet statistic based estimators mostly implemented in software enable to count or approximate either the packet reception ratio (PRR) or the average number of packet transmissions and retransmissions (RNP) before its successful reception. The problem is that these estimators need a long history to be precise and perform well for pointing out high quality links, leaving the potential of links in the transitional region aside. A summary of all presented link quality estimators can be seen in Table 2.1

| Metric | Software/Hardware | Location | Direction |
|-----------------|--------------------------|-----------------|------------------|
| RSSI | Hardware | Receiver | downlink |
| LQI | Hardware | Receiver | downlink |
| SNR | Hardware | Receiver | downlink |
| Triangle Metric | Hybrid | Receiver | downlink |
| WMEWMA | Software | Receiver | downlink |
| ETX | Software | Receiver | bidirectional |
| FLQE | Hybrid | Receiver | bidirectional |
| RNP | Software | Sender | uplink |
| Fourbit | mainly Software | Hybrid | bidirectional |

Table 2.1: Summary of LQE characteristics

3

IEEE 802.15.4

The IEEE 802.15.4 standard [32] defined the physical and MAC layers for a LR-WPAN. Its main goal was to create a low data rate protocol for low power applications. The Zigbee Alliance built this on top of the IEEE 802.15.4 protocol by further defining the higher layers of the stack and releasing the Zigbee protocol.

3.1 IEEE 802.15.4 Standard

The IEEE 802.15.4 standard supports 2 different network topologies that are useful for a wireless network: the star topology or the peer-to-peer topology. In the star topology a single node is selected to be the Personal Area Network (PAN) coordinator. All other nodes associated with the network must communicate through the coordinator. The PAN coordinator may be a node with more computing resources and may be mains powered. The participating nodes are likely to be battery powered. Such a setup would be useful in home automation applications where there is a central control point. In the peer-to-peer topology any node can communicate with any neighboring nodes within reception range. There is still a PAN coordinator, but the peer-to-peer model allows for more complicated mesh network topologies. A peer-to-peer mesh would be useful in more spread out environments, such as industrial production, inventory tracking etc. Three bands are defined for the IEEE 802.15.4 band:

- 868 MHz (Europe)
- 915 MHz (North America)
- 2.4 GHz (World-wide)

| Data symbol (decimal) | Data symbol (binary) ($b_0 b_1 b_2 b_3$) | Chip values ($c_0 c_1 \dots c_{30} c_{31}$) |
|--------------------------|--|--|
| 0 | 0000 | 11011001110000110101001000101110 |
| 1 | 1000 | 11101101100111000011010100100010 |
| 2 | 0100 | 00101110110110011100001101010010 |
| 3 | 1100 | 00100010111011011001110000110101 |
| 4 | 0010 | 01010010001011101101100111000011 |
| 5 | 1010 | 00110101001000101110110110011100 |
| 6 | 0110 | 11000011010100100010111011011001 |
| 7 | 1110 | 10011100001101010010001011101101 |
| 8 | 0001 | 10001100100101100000011101111011 |
| 9 | 1001 | 10111000110010010110000001110111 |
| 10 | 0101 | 01111011100011001001011000000111 |
| 11 | 1101 | 01110111101110001100100101100000 |
| 12 | 0011 | 00000111011110111000110010010110 |
| 13 | 1011 | 01100000011101111011100011001001 |
| 14 | 0111 | 10010110000001110111101110001100 |
| 15 | 1111 | 11001001011000000111011110111000 |

Figure 3.1: IEEE 802.15.4 symbol to chipsequence conversion

Different frequency regulations mean idfferent bands are available in different geographical location. The 2.4 GHz IEEE 802.15.4 band has seen major development as it is availabel world-wide. We will use this band in the experiments conducted in this thesis. The 2.4 GHz band contains 16 channels that start at the channel number 11 and ent at the channel number 26. The channel numbers 0 to 10 are allocatet to the 868 and 915 MHz bands. Channels are spaced 5 MHz apart with a spectral window of 2 MHz. The center frequency of each channel number k is defined as follows:

$$F_c(k) = 2405 + 5(k - 11)MHz, \forall \in 11, \dots, 26$$

In the 2.4 GHz band, data is transmitted at a rate of 250 kbit/s. Transmitted data is first converted into 4 bit data symbols, which are then spread according to a predefined spreading sequence to a 32-bit chip sequence at a rate of 2 MChips/s. The different spreadign sequences are listed in Figure 3.1. The chipping sequence is then modulated using Offset-Quadrature Phase Shift Keying (O-QPSK) and the resulting signal is sent out centered at the channel frequency. A Physical Protocol Data unit (PPDU) frame is structured as in Figure 3.2. Within the PPDU there is a synchronization header, a frame length field and then the MAC Protocol Data Unit (MPDU). The frame length field is composed of 7 bits, meaning that IEEE 802.15.4 packets have a maximum MPDU size of 127 bytes. The MPDU contains the Frame Control Field (FCF), sequence number,

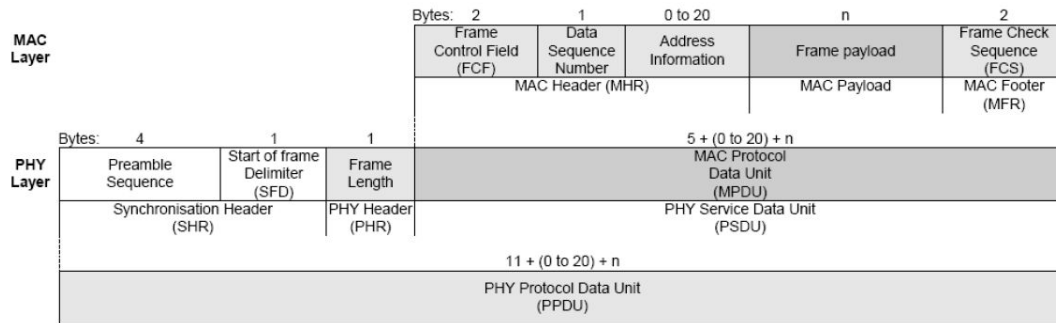


Figure 3.2: Frame Layout of an IEEE 802.15.4 Packet

address field, frame payload, and finally the Frame Check Sequence (FCS). The fields before the payload contain metadata regarding the contents of the payload. The FCS is an important first check in ensuring the integrity of the data. Protocols higher up in the stack may make extra checks for data integrity, but at the lower level it is implemented as the 16-bit CRC checksum of the MPDU.

3.2 ZigBee

The Zigbee Alliance [33] aims to create interoperable products for home and industrial use. With a built in Zigbee device these products can transmit sensor data, sensor health, commands, or updates wirelessly. By following the Zigbee standard, devices from different manufacturers will be able to inter-operate. One example involving ZigBee Technology is a home control center [34]. The center console aggregates sensor data of room illumination, humidity, temperature and air movement. Using the gathered data, the home control system can automatically control the lights, blinds, and air conditioning to optimize the home environment. This could be used in future energy-efficient homes and buildings where rooms are kept at optimal lighting and temperature with the minimal amount of resources

4

GNU Radio IEEE 802.15.4 SDR

There is an open time scale gap in link quality estimation between signal power based estimation computed directly on the hardware radio chip and packet statistic based estimators. The signal power based are defined over 8 symbols on our reference chip CC2420 and the packet statistic based estimators start being useful from a 10 packet history onwards. GnuRadio is a software defined radio realization capable of collecting more information from a received packet due to the fact that usually on hardware implemented signal processing can be observed as it is implemented in software, allowing us to find a new approach on link quality estimation. We first introduce the concept of Software defined radios and continue to present the IEEE 802.15.4 implementation on the GnuRadio platform using a USRP2 hardware radio frontend.

4.1 Software Defined Radio

A Software Defined Radio (SDR) is a radio in which some or the entire physical layer functions are software defined. SDR attempts to create a flexible platform by using software instead of traditional hardware to perform operations on signals. The ideal SDR platform would use as little hardware as possible and let the software deal with all of the signal processing [35]. An ideal receiver might have just an antenna connected to an ADC. Samples would then be read from the ADC and software would handle all the signal processing. Using Software to process signals might use much more computational power than implemented on an ASIC but offers other benefits. Software is quick to compile and load, which gives iterative development a much higher cycle rate. Researchers and developers who are looking to experiment with different filter applications can quickly prototype solutions and have real world results using a SDR. Software that is developed

for signal processing can easily be shared and adapted by others allowing new projects to build upon existing code. This code sharing is easily demonstrated by the GNU Radio community, where the main part of the SDR implementation of this thesis is build from. SDR has advantages to manufacturers because they only need to produce and support one hardware platform which can cut development cost. For service providers using SDR, they can quickly update and change their network with only slight hardware changes. End users of SDR products will be able to communicate efficiently and have the most up to date features without having to buy a new hardware platform each time. The additional observation perspectives to signal processing provided by the concept of SDR promise new approaches to link quality estimation.

4.2 IEEE 802.15.4 Implementation on GNU Radio USRP2 Platform

In order to find new approaches to link quality estimation we use a setup consisting of a GNU Radio SDR running on a notebook computer connected to th new version of the Universal Software Radio Peripheral (USRP2), where a RFX2400 daughterboard is installed which can deal with the 2.4 GHz frequency band of IEEE802.15.4. On top of it we run a modified version of the UCLA ZigBee [36,37] code.

4.2.1 GNU Radio

As the definition on GNU Radio's official web site [38], "GNU Radio is a free software development toolkit that provides the signal processing runtime and processing blocks to implement software radios using readily-available, low-cost external RF hardware and commodity processors." Examples of GNU radio applications are decoding HDTV pictures, receiving and sending AM/FM broadcast radio, and there is support for some simple modulation schemes like AM/FM/PSK. A different project has implemented a IEEE 802.11b receiver capable of receiving low data rate transmission and another a GMSK receiver able to communicate with certain satellites. To achieve such applications GNU Radio provides simple signal processing blocks written in C++. By using SWIG (Simplified Wrapper and Interface Generator), an interface an interface compiler which allows integrating C/C++ into scripting languages, GNU Radio provides a simple interface to the signal processing blocks from Python scripting language. Using the power of this scripting language, signal processing blocks are simply connected and can run at native speed without interpretation. Currently GNU Radio operates on streamed date. Newer packet based communication portocols would be better suited if processed in a packetized manner, which is one of the next development goals of GNU Radio. We use the current version 3.2 which is available in a stable release for Ubuntu 9.04 Jaunty or later.

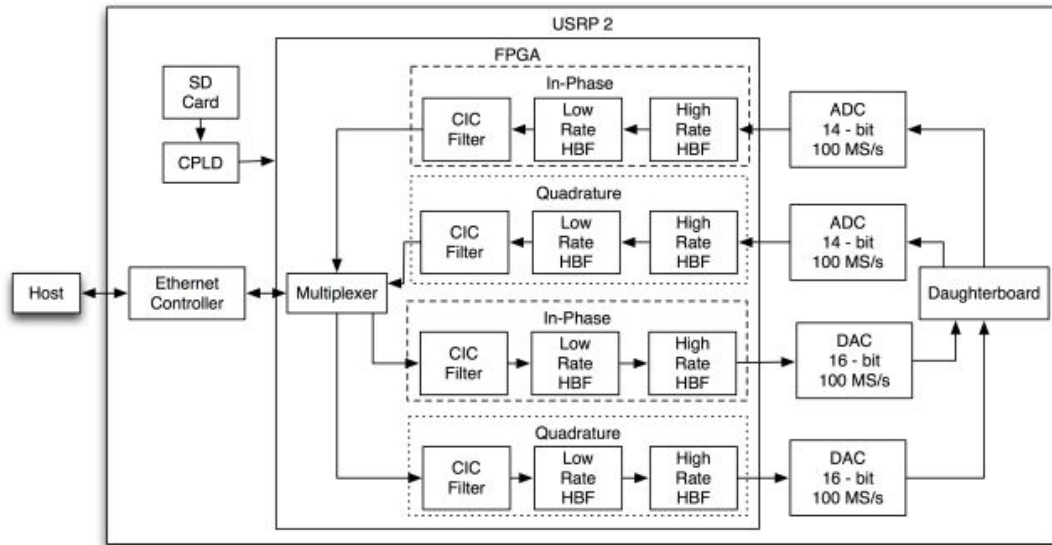


Figure 4.1: Block diagram of a USRP2

4.2.2 USRP2

GNU Radio by itself is not very useful since it still needs some hardware to interface to real world systems. GNU Radio therefore supports several different hardware platforms including soundcards and multiple different FR frontends to receive different frequency bands of the RF spectrum. The most commonly used one is the Universal Software Radio Peripheral (USRP) or the more recent USRP2. Matt Ettus [39] developed the original USRP which was meant to be a simple and flexible platform for software radios. It contains ADCs, DACs and a FPGA which can be programmed to do some basic signal processing and is used to implement the Digital Down Converter (DDC) block. The actual radio frequency (RF) frontend is represented by installable daughterboards capable of tuning to different bands depending on the daughterboard used. A daughterboard downconverts a signal to an intermediate Frequency (IF) that the ADC can handle. In our setup we use the RFX2400 which is set for frequencies from 2.25 GHz to 2.9 GHz. The USRP2 is a more powerful version of the USRP. Due to the similar design, it is still compatible with the original daughterboards. USRP2 includes ADCs capable of 14-bit 100 MS/s conversion as well as 16-bit DAC's at 400 MS/s. The best improvement is the connection between the USRP2 and the GNU Radio running host PC implemented as a Gigabit Ethernet (GigE) link instead of the USB 2.0 link available for the old USRP. GigE has a maximum transfer rate of 125 MB/s equivalent to 30 MS/s. Thus the USRP2 has a sampling window maximum of 25MHz when using a decimation rate of 4. An overview of the components in the USRP2 is shown in Figure ??

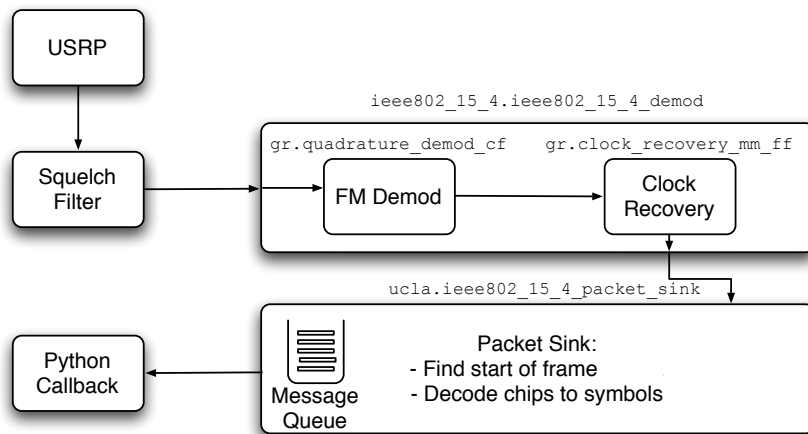


Figure 4.2: GnuRadio IEEE 802.15.4 Demodulation [36]

4.2.3 UCLA ZigBee Project

The setup of a USRP2 and the GNU Radio software still is not capable of transmitting IEEE 802.15.4 data. The UCLA Zigbee project initiated by Thomas Schmid [36] implements the missing signal processing blocks necessary for modulating and demodulating the corresponding signals.

- Demodulation:** An overview of the demodulation process is given in Figure 4.2. IEEE 802.15.4 as mentioned earlier uses O-QPSK modulation. J. Notor et al describes in [40] describes an easy way to demodulate a O-QPSK signal non-coherently to be a Low IF Receiver which actually implements a MSK demodulator, but since O-QPSK with half sine puls shapes and MSK are the same, this will work. First, the data comes from the USRP2 into a squelch filter, which can be configured to let pass only signals which have a vertain dB strength. This avoids unnecessary attempts to decode noise and allows to computer to idle between active communication. The squelch filter passes the signal into the FM demodulator ,which decodes the MSK signal. It is followed by a block recovery block, which implementas a Mueller and Müller discrete time error-tracking synchronizer. The synchronizer outputs the chipsequences ready for slicing. Physical frame detection and decoding the whole MPDU with the help or the packet length field is implemented in the packet sink C++ object. Once a complete MPDU is found, it is added to a message queue. An external python thread is observing the message queue in the packet sink. As soon as there are messages in the queue, the thread starts to process the MPDU further. In our latest experiment for example we pass the necessary content to a UDP/IP application.
- Modulation:** On the sender side we need a baseband signal fed into the USRP2. Therefore, a "O-QPSK Modulator with Half-Sine Shaping" is implemented. Fig-

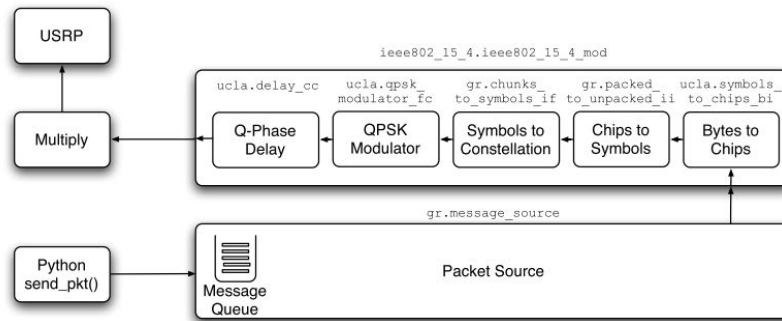


Figure 4.3: Block schema of the modulator implemented in GNU Radio [36]

Figure 4.3 shows an overview of the necessary signal processing blocks. Before the data stream can be modulated, it needs to be spread with the correct spreading sequence. First the python application puts the messages in a packet source queue. From there, the packet source generates a stream of bytes which are sent into the IEEE802.15.4 Modulation block. This block first translates the bytes into chips by spreading the byte sequence according to the IEEE 802.15.4 protocol DSSS definition. For each byte, it takes first the least significant 4-bit block and spreads it with the 32-bit spreading sequence. Then, it takes the most significant 4-bit block and spreads it again. This sequence of 64 bit is then sent to the next block as two unsigned 32 bit integers. The next block, "Chips to Symbols" translates the integers to a another sequence of integers where each bit in the integer is represented as a 0 or 1 integer, i.e., from one integer input we generate 32 output integers. The block processes the most significant bit of each integer first. Once we have the 0/1 integers, we translate them to the constellation, i.e., we map 0 to -1 and 1 to 1. The stream of -1 and 1 is then fed into the QPSK modulator which outputs the complex baseband QPSK signal with half-sine pulse shaping. To finally generate the O-QPSK signal we pass the complex baseband signal through a Q-Phase delay which delays the Q-Phase by two samples.

5

Exploration of SDR Capabilities for Link Quality Estimation

The survey on strengths and weaknesses of current link quality estimators has proven that there is still room for optimization in the dimensions of speed and accuracy. In summary, quick and agile link estimators like hardware provided signal strength measurements namely RSSI or LQI only use 8 symbols to deliver an estimate but lack accuracy, especially in complex channel conditions. On the other side, packet statistic based estimators such as ETX or WMEWMA can be very reliable in estimating static links but use a history of at least ten packets to be precise. Taking into account the capabilities of software defined radios, we start to explore possible new approaches to link quality estimation analyzing the behavior of metrics to collect during the demodulation process. We introduce five basic scenarios to represent different link conditions. On these scenarios we begin analyzing the characteristics of chip and symbol errors towards their potential of estimating PRR, introduce a digital SNR as a replacement for the unfortunately not available hardware based signal strength metrics on the USRP2. To finish off, we take a closer look at the synchronization phase of the packets.

5.1 Experimental Scenarios

A good Link quality estimator performs well in different environments and conditions. To get to know the capabilities of SDR in link quality estimation, we analyze chip and symbol errors as well as our digital SNR in five different measurement scenarios representing different link condition in which ZigBee like sensors often are distributed. As a basic an least biased case, we include a cable connection scenario additional to the

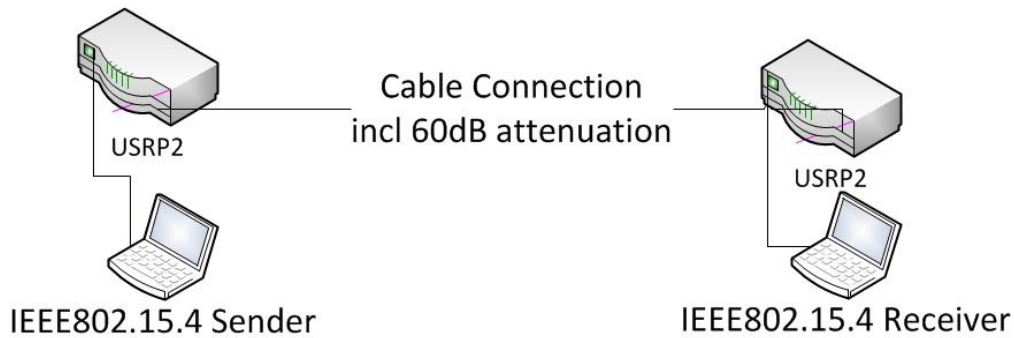


Figure 5.1: Setup cable connection scenario

wireless cases represented by an indoor line of sight (LOS), non line of sight (NLOS), outdoor and a mobile scenario. All of which will be presented in detail in this chapter. One of our goals is to find a link quality estimator which performs well throughout all channel conditions in terms of packet reception ratio. Therefore we conducted several measurement runs for each scenario with a modified transmitter amplitude to cover the whole range of link qualities. One measurement run consisted of 10000 transmitted packets of 27 bytes payload as used by Schmid in [36]. The payload consisted of a two byte sequence number and a 25 byte random payload. The packets were transmitted on channel 16 (2430MHz) at 250kbps, the common bit rate for IEEE802.15.4 connections. For every run, the sender logged every sent symbol and chip sequence in the *Symbol-to-Chip* block of the modified UCLA Zigbee implementation. Correspondingly, the receiver logged on the other side every received chip sequence and resulting symbol in the *Packet Sink* block as well as a bit earlier in the demodulation process the digital signal power values where logged in the *FM Demodulation* block to define a digital SNR. Sender and receiver code are executed on separate dual core notebooks which are connected to the USRP2 hardware radio frontends through a Gigabit Ethernet Connection.

5.1.1 Attenuated Cable Connection

The first and most basic example served to get the least biased and static insights to the behavior of our modified UCLA ZigBee implementation. Sender and Receiver were connected by a shielded 0.5m cable and a 60dB attenuator. The basic setup is presented in Figure 5.1.

5.1.2 Indoor Line of Sight

In Figure 5.2a we see our second scenario. The second measurement series was conducted on a simple indoor line of sight LOS connections. The cable was replaced by antennas on the sender and receiver side. Both USRPs were placed 2m apart on the floor antennas oriented perpendicular.

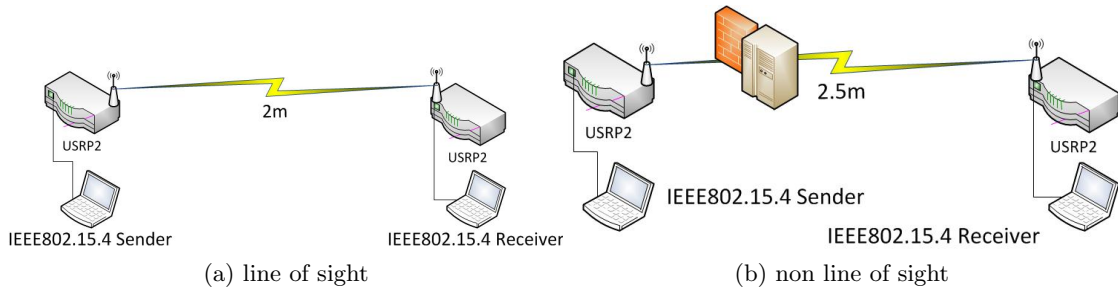


Figure 5.2: indoor office scenarios

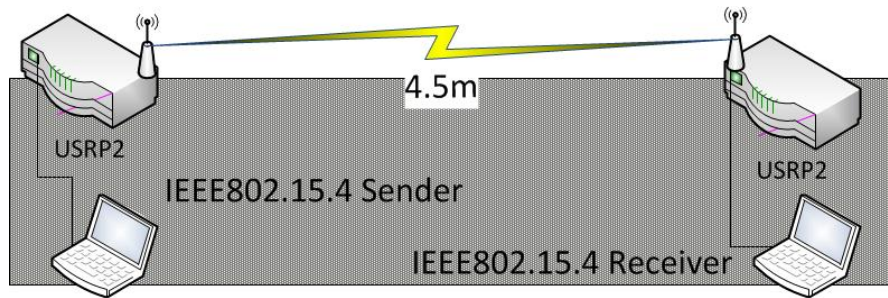


Figure 5.3: Outdoor measurement scenario

5.1.3 Indoor None Line of Sight

As a third scenario we placed the sender and receiver in positions offering only none line of sight connections. A mixture of metal cupboards and concrete pillars blocked the LOS between the two antennas as shown in Figure 5.2b.

5.1.4 Outdoor

Wireless sensors are often placed outdoors to monitor one or several environmental variables. So our fourth scenario takes place outdoors. The sender and receiver are therefore placed on a wall 4.5m apart presented in Figure 5.3. The measurement series took place at about 10° Celsius which did not influence the performance of the devices in service.

5.1.5 Outdoor Mobile

To represent a faster changing channel condition we introduce a fifth measurement scenario where we force a change of the channel condition by moving the sender around, as we could imagine by sensors worn on body and transmitting collected information to available base stations. The SDR setup including the notebook as well as the USRP2 is

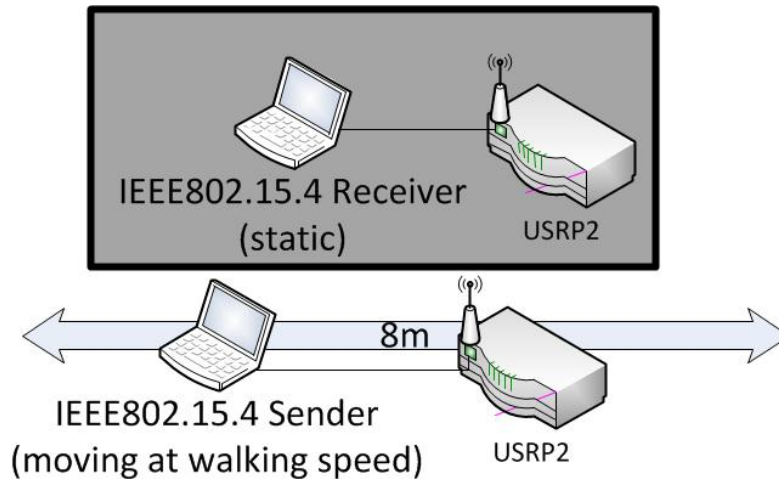


Figure 5.4: mobile measurement scenario

not so easy to move around as the USRP2 needs to be constantly connected to the mains supply. As schematically displayed in Figure 5.4 we placed all the necessary devices of our IEEE802.15.4 sender on a tray and moved back and forth approximately 8m passing by the receiver each time. During one measurement run of 10000 packets we could create 20 sequences of approaching and moving away from the receiver.

5.2 Chip Errors

Provided by the vast amount of logging data, we started programming Matlab [41] scripts to analyze the behavior of chip and symbol errors towards their correlation to the packet reception ratio. The first difficulty was to sort the logged data on receiver side according to the sequence number and position in packet to be able to compare them properly to the sent chips and symbols. Especially for bad channels some of the packets had to be discarded when there was an error in the sequence number or a lot of errors in the synchronization process which made it impossible to determine where one packet starts and the other one ends. Having solved all this problems due to the incomplete log files of the low quality channels, we could start analyze the number the position and distribution of errors occurring on chip and symbol level. We first take a closer look at the chip errors.

5.2.1 Distribution of Chip Errors

Motivated by the study of Han et al. [42] where they analyze the position of bit errors in IEEE802.11 packets and discover three different often occurring patterns, we transform their concept of analysis onto the chip errors of our IEEE802.15.4 SDR implementation. A first approach was to look at the position of chip errors in a symbol. A representative

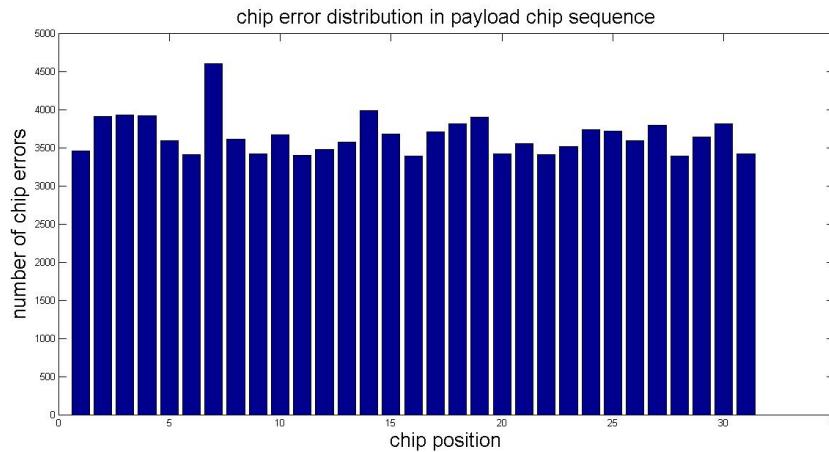


Figure 5.5: Chip error distribution in a payload symbol

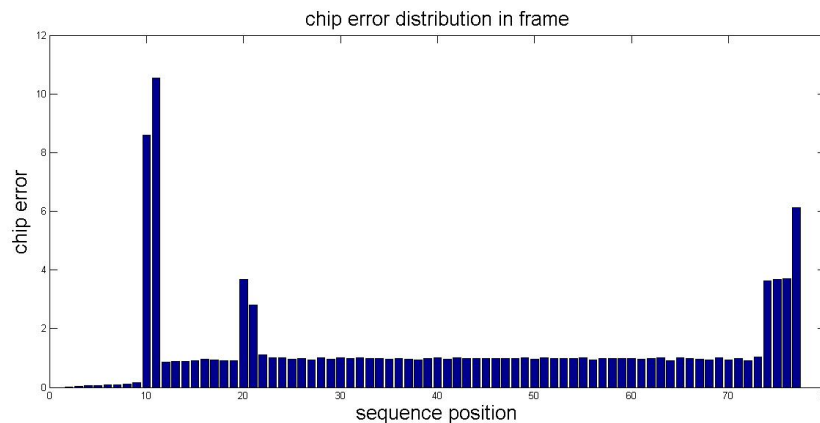


Figure 5.6: Chip Error Distribution in one Packet

plot of such an analysis is presented in Figure 5.5.

Observing these results over all measurement series of all scenarios we could exclude that chips at a certain position are erroneous significantly more often. Moving up one level we sum the chip errors of one symbol and determine if some chip sequences in a packet show a significantly higher number of chip errors. As we see in Figure 5.6 there are some characteristic peaks at three locations. The first two high bars are located at position of the SFD symbols the next two at the position of the sequence numbers and the last four represent the CRC. All these peaks are due to the way IEEE802.15.4 demodulation is implemented in GNU Radio and do not state important information about link quality of the wireless channel. This is explained in more detail in the symbol error section.

Therefore we zoom out once more and take a look at the summed up chip errors in a whole packet. In Figure 5.7 we see a plot of a whole LOS measurement series. The

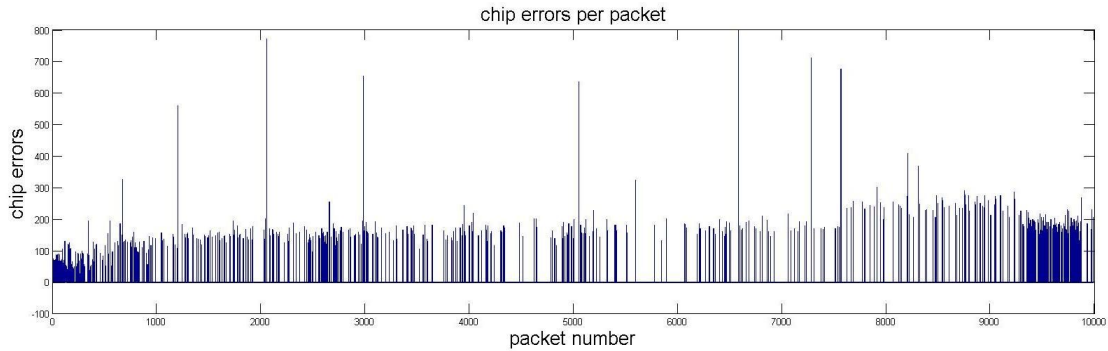


Figure 5.7: Chip Error per Packet in a Measurement Series

whole run had a packet reception ratio of 10 %. The bars represent the number of chip errors in a packet. If there is no bar, no packet was successfully received. What we observe in general is that where the white area gets more dominant the chip errors per packet increase as well. This is to be observed in all five scenarios, so we assume that the number of chip errors in a packet can be an indicator of current channel quality. The few outliers of significantly higher chip error counts occurred in packets bearing symbol errors in the payload which results in wrong CRCs and therefore these packets are discarded by higher layer applications.

To further assess this assumption gained observing the behavior of chip errors per packet we averaged the local chip errors to a value representing the chip errors of one DSSS frame which represents a symbol and compared it to the current PRR of a sliding window of 100 packets around the one we computed the chip errors. Our assumption was confirmed by correlation plots such as Figure 5.8 where we clearly see that the chip errors behave linearly reciprocal to the packet reception ratio. The different colors represent measurement runs conducted setting different transmitter amplitude values to produce low quality channels.

Wu et. al. took a look at patterns of chip errors per symbol in [43] and used these patterns to determine if a sensor was moving in [44]. Producing similar plots shown in Figure 5.9, we did not observe all the same patterns they did. The biggest difference in their approach to ours is the fact they compared their received chip sequences to a best match sequence and not to the actual corresponding sent sequence.

5.2.2 Burstiness of Chip Errors

In [6] Srinivasan et. al. introduced the β -factor as measurement parameter of link burstiness. It tries to describe if packet losses happen in bursts or almost equally distributed based on conditional probability delivery functions. We apply their concept for our purposes on chip level to determine if what is valid for packet errors also counts for chips. An example of a conditional probability delivery function (CPDF) for chips is presented

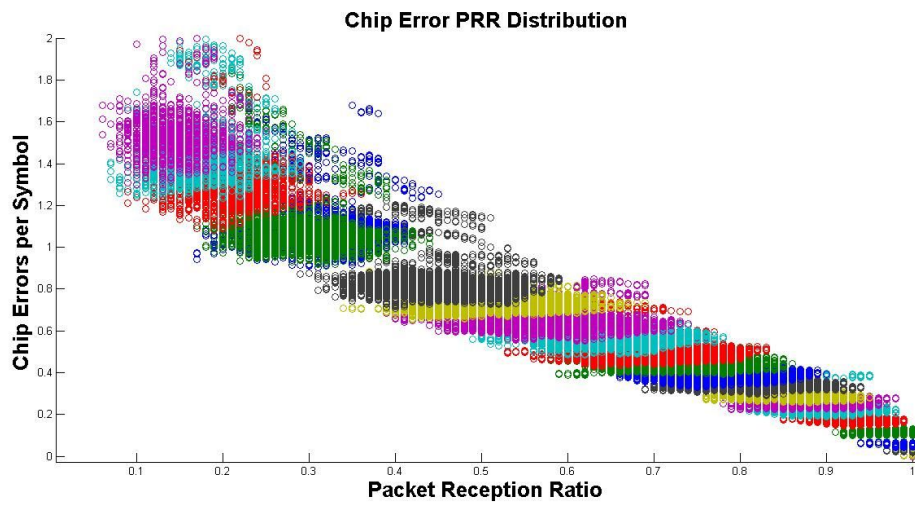


Figure 5.8: Chip Errors per Symbol in Payload Distribution

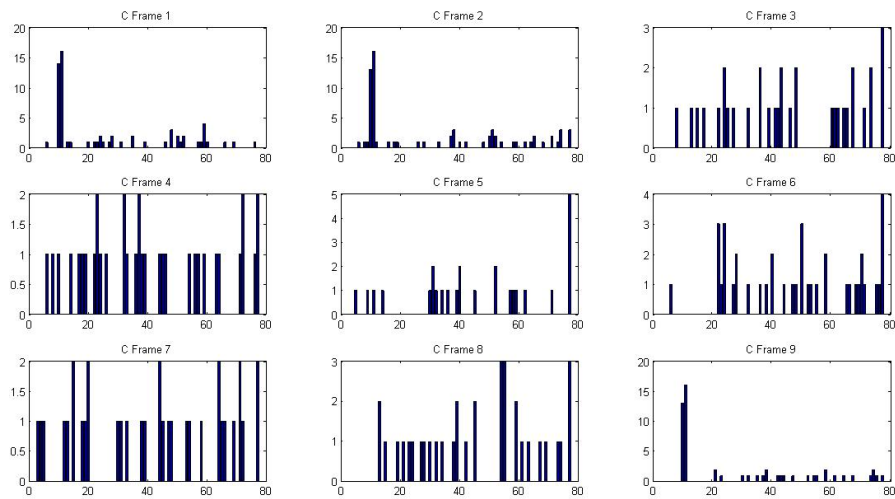


Figure 5.9: Chip Error distribution of 9 consecutive Symbols

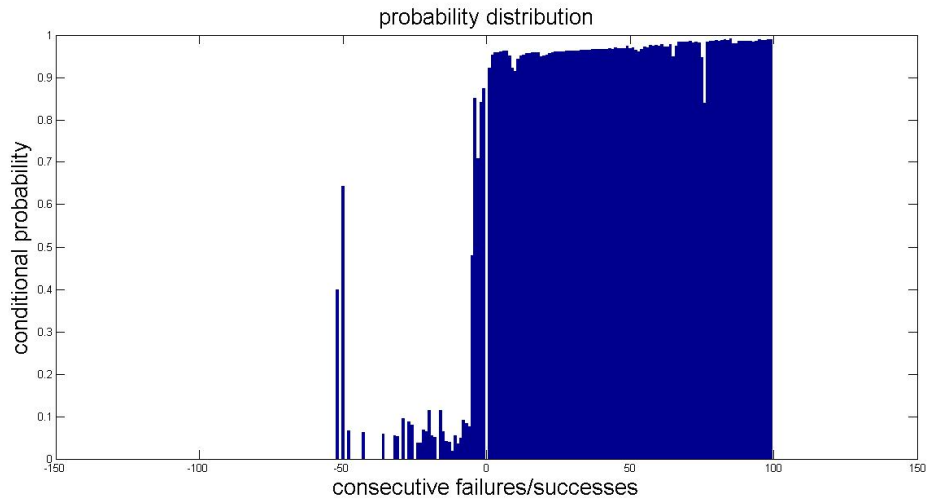


Figure 5.10: Conditional Probability distribution of received chips

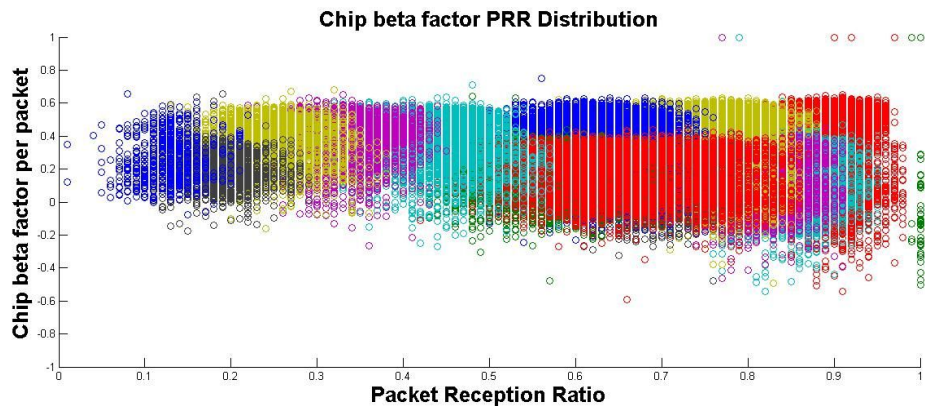


Figure 5.11: Chip Beta factor distribution

in Figure 5.10.

Based on these CPDFs we computed the beta factors for the measurement series and tried to figure out, if it could be another indicator of link quality. In Figure 5.11 we see that the chip β -factor does not correlate in a usable way to link quality in terms of PRR. We further tried to combine the chip β -factor with the chip errors per symbol in each packet but found no improvement over only using chip errors per symbol as link quality indicator.

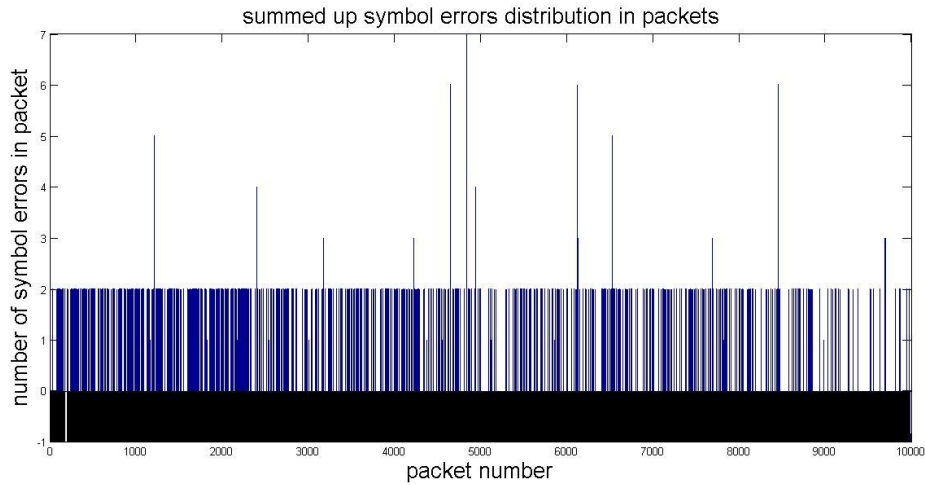


Figure 5.12: Symbol Errors during a Measurement Run for a 10% PRR channel

5.3 Symbol Errors

An abstraction level higher we observe symbol errors. One symbol representing four data bits in our case. As for the chip errors we study the number position and distribution of the symbol errors.

5.3.1 Distribution of Symbol Errors

In the *Packet Sink* signal processing block of the UCLA ZigBee implementation, we decide for a symbol in a best match case, so only a few chip errors can already be corrected and we therefore lose some possible useful information about link quality. Symbol errors for a 10% PRR channel is presented in Figure 5.12. We observe that even for such a low quality channel, symbol errors only occur sparsely. If we only consider the payload, symbol errors are almost totally reduced due to filtering out incompleteness during synchronization. The few outliers we see cause at the same time packet loss due to CRC incorrectness. In comparison to the packets not even captured at the receiver side CRC mismatch cases only contribute in the range of 10^{-3} to the total amount of packet loss and is therefore neglectable.

5.3.2 Burstiness of Symbol Errors

β -factors determined based on symbol errors as a measure of link burstiness does behave similar as it is the case for chip errors. In Figure 5.13 we have a representative summary of the LOS scenario showing the probabilities of successfully received chips and symbols as well as the β -factors for all three abstraction levels. What we get out of these plots is that we see, that only the probability of chip correctness decreases according to the

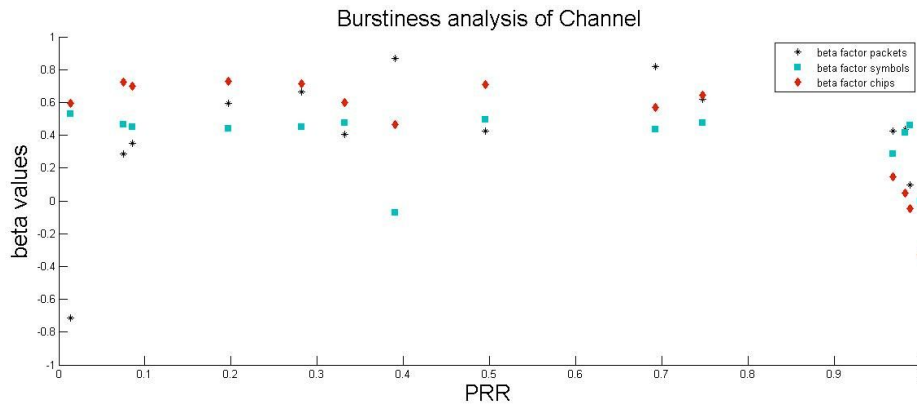


Figure 5.13: Comparison of beta-factors for different Link Qualities

channel quality. The packet level β -factor behaves as expected and is highest for channels in the transitional region and decreases towards the edges of perfect and broken channels. The same does not apply for on the symbol and chip level and will therefore not be used as an indicator for link quality.

5.4 Digital SNR

Our UCLA Zigbee Implementation intends to achieve the same IEEE802.15.4 performance as the CC2420 radio chip [22] produced by Chipcon. In Chapter 2 we showed that quick link quality estimation for IEEE802.15.4 based networks often rely on the hardware provided RSSI and LQI values. Unfortunately, the current GNU Radio release is not able to read out the RSSI value of the daughterboard card installed in the USRP2 hardware radio frontend. To have a value for comparison based on signal strength, we create our own digital SNR from signal samples transmitted from the USRP2 to the GNU Radio Notebook. This is done in the earliest step of demodulation at the beginning of the *FM Demod* signal processing block specified in *quadrature_demod_cf.cc*. We calculate the signal power of the received samples and log them with a corresponding time stamp, So the analyzing Matlab scripts can assign the values to the corresponding packets received. Through the timestamps, we are able to assign packets to their corresponding signal strength values and at the same time, we can determine the noise floor between two packets. Collecting this information, we compute our software defined SNR in the way of [45] subtracting the current noise floor from the signal strength of each packet. In Figure 5.14 we see displayed the behavior of our representative 10% LOS channel used as example before. Only a slight tendency is visible for the SNR to drop when more packets are missed, which is consistent to the observation of the transitional region of earlier research as in [14].

Measuring this implementation specific software defined SNR we observe in Figure 5.15 how this value correlates to the PRR of a local window of 100 packets around the mea-

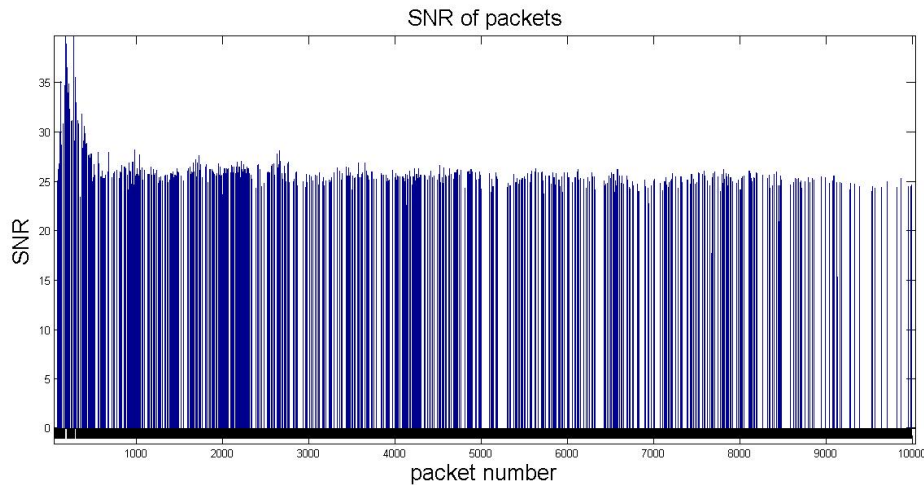


Figure 5.14: SNR behavior during one measurement run

surement. As we see we could reproduce the steep characteristics of the transitional region where the channel quality drops from perfect to broken in only a few dB. Comparing Figure 5.15a and Figure 5.15b we notice that for the cable connection example on the left, we could still see some linear behavior traveling the transitional region as for the LOS antenna connection scenario on the right we get a much more blurry result.

Applying the concept of our software defined SNR we state that it is a well chosen replacement for real hardware provided values to be able to include knowledge of signal strength values to model link quality estimators.

5.5 Synchronization Errors

Considering chip errors and the digital SNR provides already a lot of new insights to link quality but is only useful assigned to a specific packet. As we have described earlier, almost all packet losses are caused by not even being able to synchronize correctly and not because there would be too many chip errors in a packet. So as a logical next step we assume to get more information about channel quality by analyzing the synchronization phase of a packet. IEEE 802.15.4 packets start with 8 zero symbols followed by the start frame delimiter (SFD) composed of the symbols 10 and 7. The implementation of this synchronization phase is programmed in the *packet sink* signal processing block as a finite state machine. The implementation requires at least one zero symbol and the SFD to be detected by the receiver to continue decoding received chips to symbols of the PPDU. So we studied the logfiles more intensively during the synchronization phase and detected two more possible link quality indicators. The first indicator is simply the chip errors per symbol during the synchronization phase while the second indicator counts the number of synchronization zero symbols which were successfully received. In the

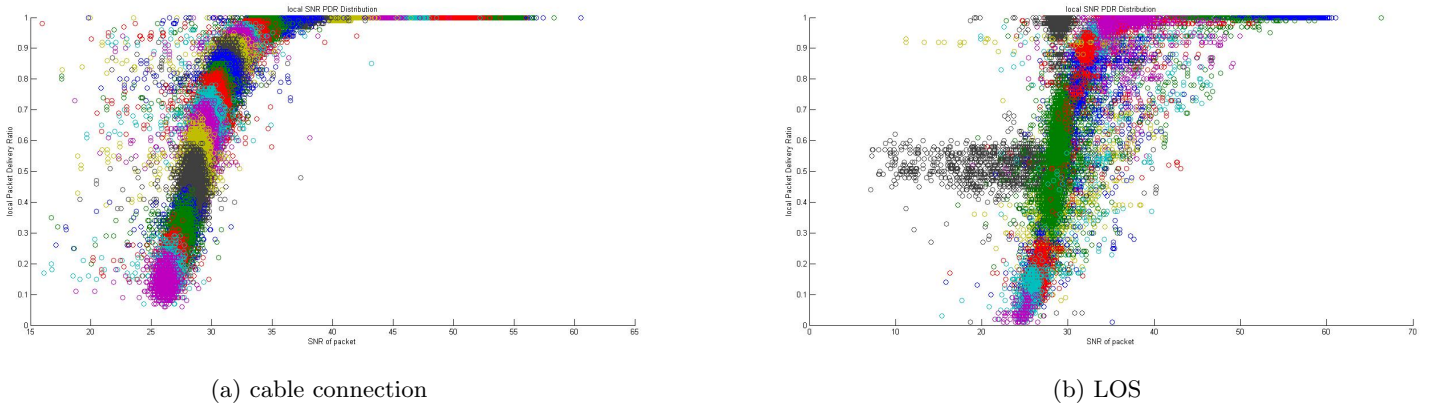


Figure 5.15: Digital SNR Distribution

following experimental runs we logged this indicators together with a time stamp so we could even define these indicators when packets were not successfully received and only some symbols from the synchronization phase were received, which would be a great advantage over packet statistic based metrics.

Correlation of chip errors per symbol compared to PRR shows similar characteristics as it was the case for the whole payload as presented in Figure 5.16. What we could gain using this indicator would be the additional information about packets not managing to pass the synchronization phase successfully. The basic example plot is based on a cable connection measurement providing static characteristics necessary to evaluate the indicators without interference and fading effects. Looking at the correlation of the second indicator, the number of zeros successfully received during synchronization phase as displayed in Figure 5.17 we see our assumption confirmed that the better the channel in terms of PRR, the more synchronization zeros are successfully received. Further interesting is the fact that the values of this indicator cover the range from 0 to 8 as chip errors in general are located between 0 and 2.

Unfortunately one possible advantage of analyzing the synchronization phase was not observed during the measurements on different scenarios. Only a few indicator values of a ratio of 10^{-3} more compared to number of values generated by the totally received packets could be detected even for the low quality channels. This means if the channel conditions are well enough for at least one of the synchronization zeros to be detected successfully, meaning being decoded below the implemented threshold of one chip error, the whole packet is decoded successfully. For future research it would be interesting to allow more and more errors during the synchronization phase, which could provided more information for Link quality estimation, but produces much more demodulation and decoding overhead where nothing is to be demodulated which would increase the energy consumption of the receiving device.

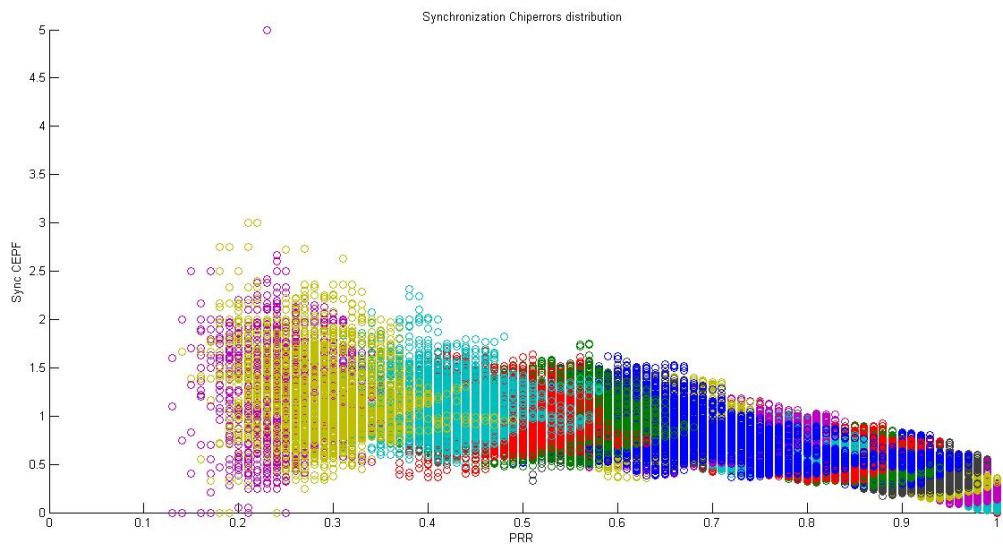


Figure 5.16: Chip Errors per Frame during Synchronization Phase

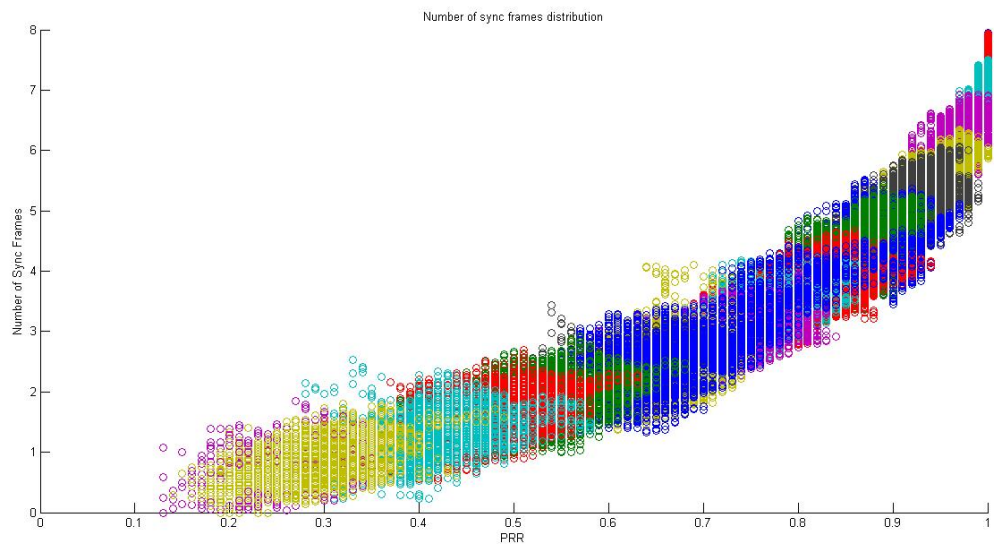


Figure 5.17: Number of Synchronization Zeros detected

5.6 LQE Potential of SDR provided Information

As the synchronization phase measurements did not provide a significant improvement over simple chip error measurement of the payload. We focus from now on the chip errors per symbol in the payload of a packet and use the software defined digital SNR as a comparison value for signal power. Based on the analysis of the of the presented metric candidates mentioned in the above sections, we state the following hypothesis to investigate.

Hypothesis 5.6.1 *Analyzing the Chip Errors during the process of demodulation, the link quality of a DSSS System can be estimated more reliable than based on Signal Power and in less time than based on packet statistics.*

6

New SDR Defined LQE Models

Using all the findings generated in Chapter 5, we take the values to define specific metrics produced by the SDR in the first section. Having defined our metrics we start creating link quality models which state how the newly defined metrics are fed into a Link Quality Estimator are processed to inform an overlaying application about the Quality of the Wireless link in use in the second section. The third section then is devoted to analyze how these models perform in our experimental scenarios.

6.1 Metric Definition

The GNU Radio SDR realization of IEEE 802.15.4 provides us with additional information about the chip error rate and a digital computation of the SNR grabbed from USRP fed samples before they are being demodulated. This allows us to define the following metrics.

6.1.1 Chip Errors per Symbol (CEPS)

An important step in demodulating DSSS signals is assigning the right corresponding symbol to a received PN-sequence. This is done choosing the 'best match' PN-sequence. To decide which of the 16 symbols is the best match is done using a maximum likelihood decoder (MLD). The received sequence of 32 chips R is compared to the 16 predefined PN codes S_0, S_1, \dots, S_{15} from Figure 3.1 to find out a $i \in 0, 1, \dots, 15$ such that

$$CEPS = \arg_i \min h(R, S_i)$$

where h is the hamming distance of two chip sequences (the number of positions containing different chips). Given the maximum likelihood match i , we define the value of $h(R, S_i)$ as *Chip Errors per Symbol*. In our initial experiment, we could even compare R to the actually sent chip sequence due to our complete log files.

6.1.2 Digital SNR

Due to the fact, that we unfortunately cannot read the physical received signal strength indicator (RSSI) value from the USRP2, we helped ourself by extracting a digital signal power value out of the signal processing block *quadrature_demod_cf* where the demodulation process of the received USRP2 samples starts. We define our digital signal to noise ratio to be

$$dSNR = \frac{\text{average signal power of a packet}}{\text{noise floor around packet}}$$

So we defined two new metrics extracted from the demodulation process of the GNU Radio IEEE 802.15.4 implementation. The *digital SNR (dSNR)* is a reference value right at the beginning where the SDR starts interacting to the hardware frontend and the *Chip Errors per Symbol (CEPS)* are captured just before the demodulated symbols are fed back to the higher layer post processing.

6.2 Modeling Link Quality Estimators

Having defined the two new metrics made possible using software defined radios, we start creating models converting the metrics into an expression which states the quality of the analyzed radio link. As there is no paramount metric describing link quality, we model our link quality estimators to provide an estimate of the packet reception ratio, as this value directly correlates to how much information can be sent over a certain link. We start modeling estimators based on the chip errors per symbol (CEPS) metric, continue defining dSNR estimator models and finally try to combine the two metrics in a third estimator model. To get a fair assessment of the estimation precision we use a sliding window approach. We always let the models predict the next packet and compare this estimation precision to the actual captured packet reception ratio.

6.2.1 CEPS based Models

First, we start modeling based on chip errors per symbol. As we have seen in Figure 5.8 the correlation of chip errors to the packet reception rate as main link quality indicator shows nice linearity. So in a first step we create models using linear regression and further apply a geometric approach setting a linear model in the center of the scatter plots.

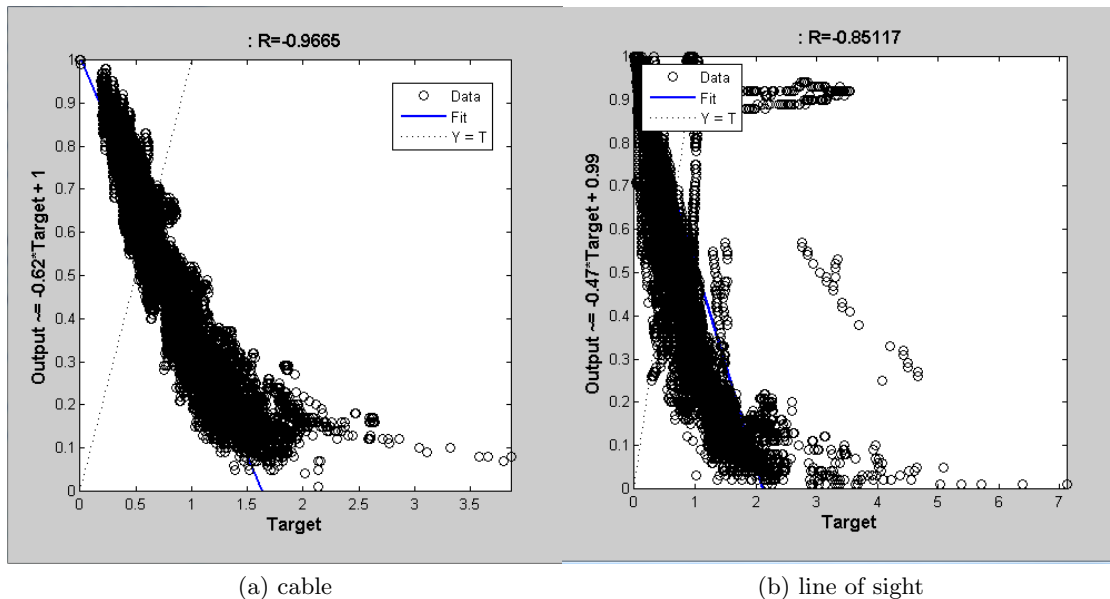


Figure 6.1: Linear regression plots of cable and line of sight measurements

Linear Regression

According to the linear characteristics of the scatter plots in Chapter 5 a logic approach is to perform a linear regression on the logged data. A linear regression creates model of the following form:

$$PRR = m * CEPS + OFFSET$$

In Figure 6.1 we see two examples of such a linear regression. The black bullets represent the data gained from the measurements and the blue line is the created linear model. The method to create the model is called *least-squares fit*, which builds the model in a way, that minimizes the squared errors of the model generated value compared to the measured real world data. The consequence of this method is that in sections on the PRR scale, where a lot of data was measured, there the model is more precise than where the data are more sparse. This is commonly the case for good quality links as you get more data when you receive more packets. To reduce this bias as seen in Figure 6.1b we introduce simple geometric models circumventing to concentrate on bulky datasets.

Simple Geometric Linear Models

Linear regression is a nice way to compute models from measurement data. It is though computationally expensive and locally biased where a more data is available. The nicely displayed linear characteristics of the chip errors in comparison to the packet reception ratio can directly be used to build simple geometric models drawn in the center of the

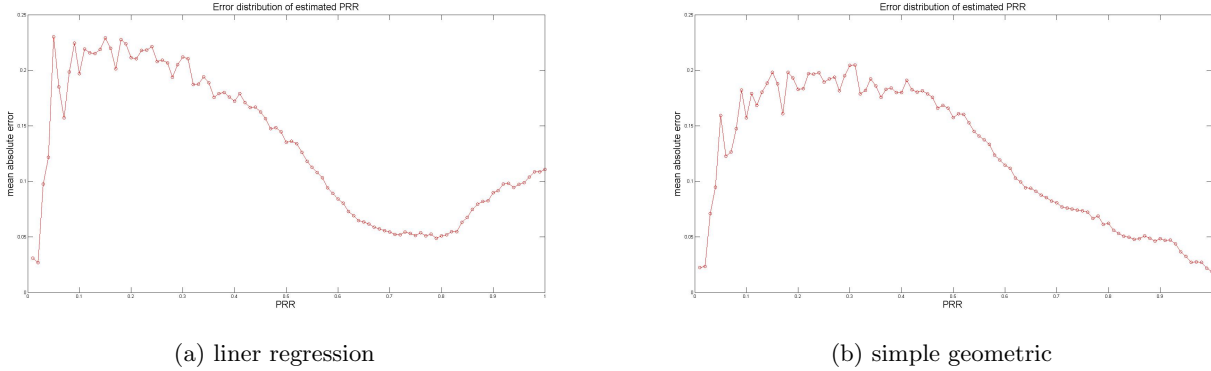


Figure 6.2: Precision of Estimation for linear regression and simple geometric models

scatter plots. Pursuing this approach we get models of the form

$$\begin{aligned}
 PRR_{est} &= 0 \quad , \quad CEPS > Chiplimit \\
 PRR_{est} &= 1 - \frac{CEPS}{Chiplimit} \quad , \quad 0 \leq CEPS \leq Chiplimit
 \end{aligned} \tag{6.1}$$

Instead of calculating time consuming lest-square fits as it is done performing a linear regression we only have to set the value $Chiplimit$ according to the displayed data. $Chiplimit$ is the value where a simple linear model intersects to the PRR axis. An example of the consequences choosing one or the other approach is presented in Figure 6.2 where we compare the absolute error over the whole range of link qualities from zero to one. The first point one observes is the fact that due to the offset the linear regression introduces we allow an small error at the borders only to improve the estimation precision between 0.6 and 0.9 PRR where we measured the most data. At the same time this is the only range where the linear regression model is more precise than the simple geometric one.

We can conclude that it is not necessary to perform a linear regression on the measured data sets. It is sufficient to draw a simple to linear model in the center of the captured values. This statement is further supported by the fact that for the most detailed and fairly distributed measurement, we find the linear regression and simple geometric model to be almost identical.

Worst Symbols Approach

Chip errors per symbol is a metric with a potential range of 0 to 32. But conducting our experiments, we mostly only observed values from 0 to 2 even for the worst channel quality's ($PRR \leq 20\%$). To improve this range we take a look at models only considering the worst symbols of the payload. This means if we look at a cumulative distribution

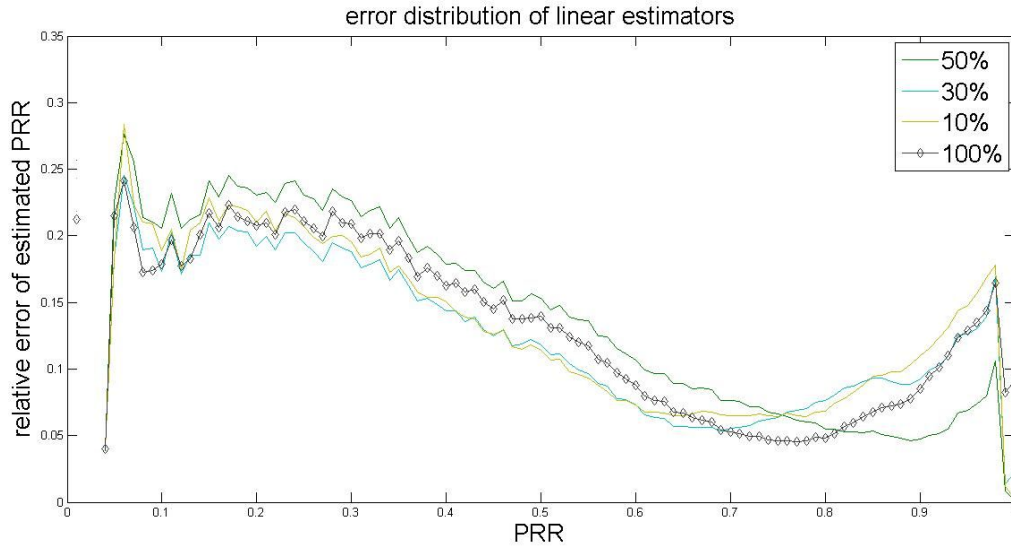


Figure 6.3: Comparison of worst symbol model performance

function of all the occurring realizations of the *CEPS* value in one sent packet, we take only a percentage containing the highest values i.e. the 20% symbols containing the most chip errors. The high value selection enables us to push the *Chiplimit* value from Equation 6.1 from 1.7 to approximately 8. The deviations around the model unfortunately stay in the same range for considering only the worst symbols. As we see in Figure 6.3 the precision of the resulting models only marginally improved or is in some areas of PRR link quality even worse than the computationally easy way of considering the whole payload for estimation.

How many Symbols are needed to estimate Link Quality

So we have seen in the last subsection that extensive sorting of the received *CEPS* values does not offer a substantial improvement. If we do not have to sort symbols to estimate link quality, we can try to improve our estimator in terms of speed. We ask ourselves how many symbols do we have to consider to get a sufficiently precise estimate of the link quality.

To determine the influence of the number of symbols we consider for our estimator, we took the most reliable values from the cable connection experiments and applied the estimation model on different numbers of symbols taken at the beginning of the payload. In Figure 6.4 we see the result of this comparison. We analyzed the influence in 20% steps in terms of PRR link quality. It is obvious that already four symbols provide an estimate in a range of 1% to the estimate generated considering the whole payload. Only for the channels ($0% < PRR < 20%$) we see an improvement in precision of 3%. Conclusively we can say, that data packets or beacon as small as two bytes of payload

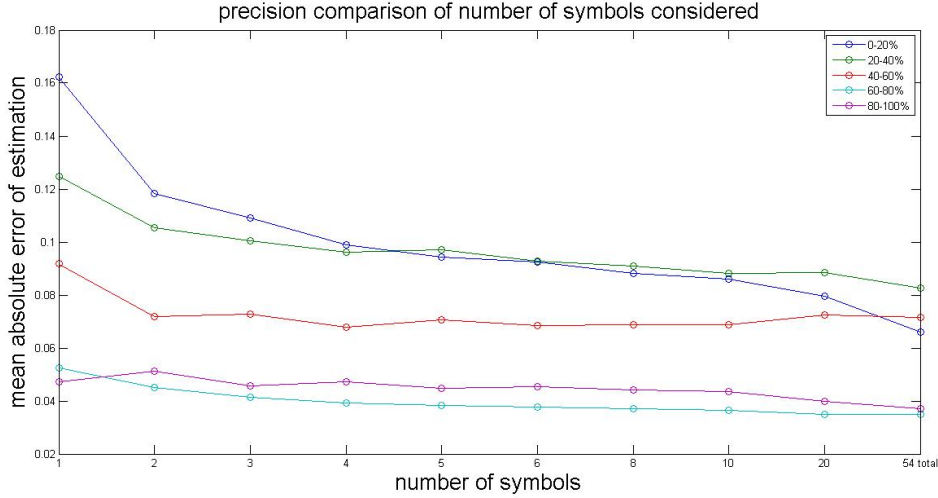


Figure 6.4: Number of Symbols Precision Trade off

are sufficient to get a qualitatively good estimate of a CEPS based estimator.

6.2.2 Digital SNR based Models

The digital SNR behaves as expected similar to hardware extracted analogue counterpart. As we see in Figure 5.15 there are large sections where the channel is either perfect or broken. In between this two regions we have the interesting narrow transitional region which bares the biggest potential for link estimator but at the same time presents the biggest challenge. So we again focus our estimators on the transitional region and neglect perfect and broken links.

Linear Model

The mentioned scatter plots showing the correlation between PRR link quality and dSNR values suggests a linear approximation of the transitional region. This leads to continuous function similar to the membership function for the ASNR membership function of the Fuzzy LQE [29].

$$\begin{aligned}
 PRR_{est} &= 0, & dSNR < t_{lower} \\
 PRR_{est} &= \frac{dSNR - t_{lower}}{t_{upper} - t_{lower}}, & t_{lower} \leq dSNR \leq t_{upper} \\
 PRR_{est} &= 1, & dSNR > t_{upper}
 \end{aligned} \tag{6.2}$$

Applying this model to the collected dSNR values we can compare the different PRR_{est} to the actual values extracted from the packet statistics. Our experiment revealed t_{lower}

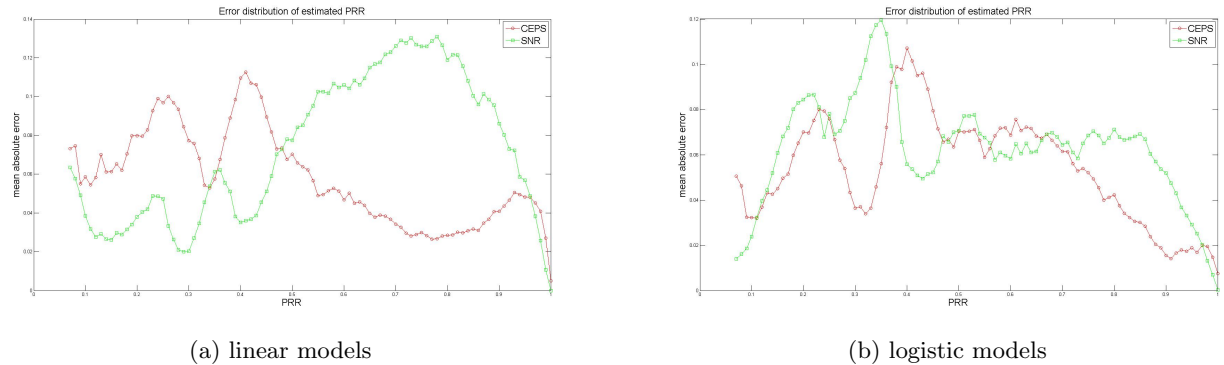


Figure 6.5: Comparison of estimation precision of the dSNR and CEPS models

to be 26 and t_{upper} to take the value 34. In Figure 6.5a we see that the absolute error of the dSNR model in the reliable cable connection does not exceed 13% over the whole range of PRR link qualities. In comparison, we see the performance on the same set of experiments for the CEPS based model.

Logistic Function

A more sophisticated approach to define models based on our measured experiment data is to use curve fitting with different functions. MATLAB provides the *curve fitting toolbox* to create appropriate models. The available functions of the toolbox did not fit our needs as the $dSNR$ but even the $CEPS$ measurements showed characteristics of the logistic function [46] defined as

$$PRR = A * \frac{1}{1 + e^{-B*(dSNR-C)}}$$

The resulting model is displayed in Figure 6.6 which pays more attention to the smooth approaching of the perfect channel. So we can reduce the absolute error of such an estimator in the section $PRR > 0.6$ but at the same time get less precise in the lower link quality section as seen in the comparative plot of Figure 6.5b. In conclusion we see that it is possible to create more accurate models using the potential of curve fitting. But the additional computational load is in no relation to the small number of percentage points we gain in only some sections of link quality. We further recognize that the two SDR defined metrics outperform each other in different sections of link quality. Inspired by known attempts to combine different metrics to improve overall estimation as in the Triangle Metric [30] FourBit [2] or Fuzzy-LQE [29] we do the same using our newly defined metrics in the next subsection.

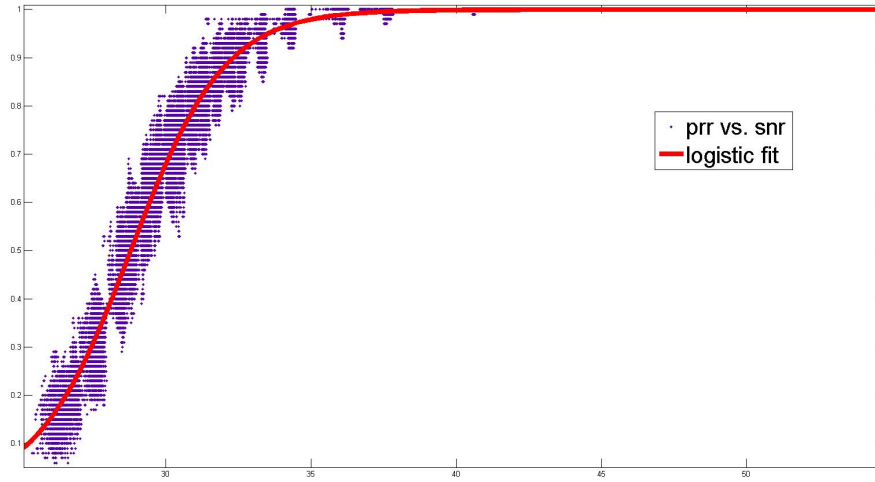


Figure 6.6: Logistic model through curve fitting

6.2.3 Combined Metric Models

CEPS and dSNR based link quality estimators as presented above already show reasonable estimation precision depending on information of only one packet. We now try to combine them to further be more equally precise over the whole range of link qualities and at the same time avoid the spikes the single metric models produce in some places. Based on our observations we obviously start with a linear combination of dSNR and CEPS as both of them could be already modeled linearly individually. Remaining open to different approaches we try a $1/x$ combination in a next step.

Linear Combination

The higher the dSNR value of a packet, the better the link quality. High CEPS values mean more chip errors and indicate therefore a worse channel. To linearly combine the two metrics, we best subtract the CEPS values of the dSNR. As both metrics are generated using the capabilities of software defined radios, we call the combined metric **SDRLQE**

$$SDRLQE = dSNR - m * CEPS$$

The scaling factor m is used to weigh the two metrics equally. In our experiments dSNR values crossed the transitional region in $8dB$ while the CEPS covered a range of 2 considering the whole payload. In this case m was set to 4.

In Figure 6.7a we observe the resulting correlation between the linearly combined SDR-LQE metric and the basic PRR link quality. The advantage of the newly defined metric

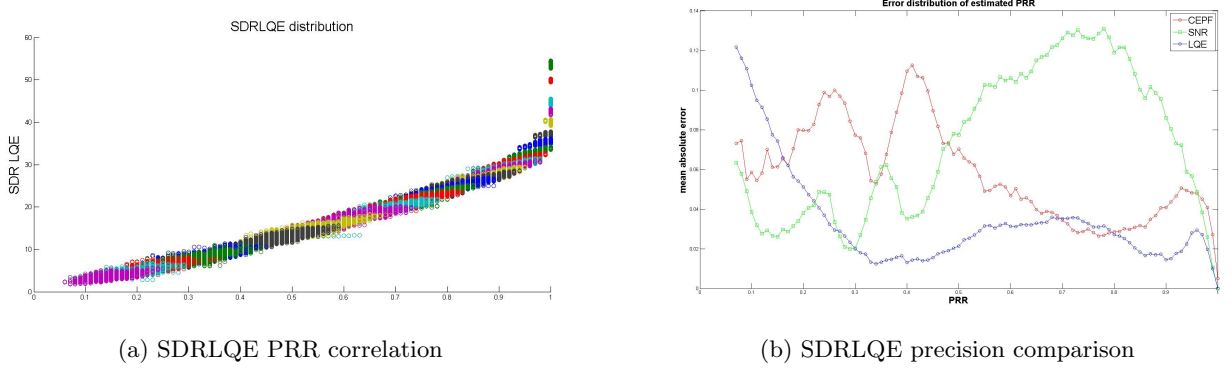


Figure 6.7: Correlation and precision of SDRLQE

is the values now cover a wider range in narrower characteristics. To compare the estimation potential of the combined metric to the single metrics *CEPS* and *dSNR* we model a linear estimator almost identical to the *dSNR* based one.

$$\begin{aligned}
 PRR_{est} &= \frac{SDRLQE - t_{lower}}{t_{upper} - t_{lower}} & , & \quad t_{lower} \leq SDRLQE \leq t_{upper} \\
 PRR_{est} &= 0 & , & \quad SDRLQE < t_{lower} \\
 PRR_{est} &= 1 & , & \quad SDRLQE > t_{upper}
 \end{aligned} \tag{6.3}$$

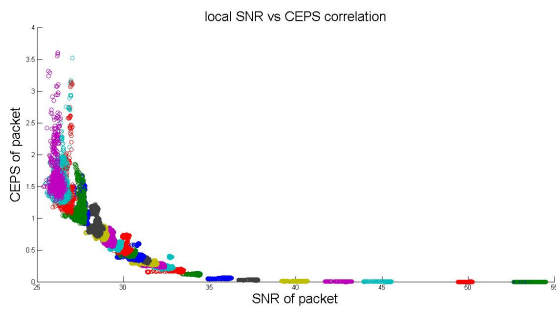
What changes is simply the values of t_{upper} and t_{lower} . The estimation performance for our basic data extensive cable connection case is as expected mostly better than the single metric based models does not include any particular spikes in the whole link quality range. Only in for the lowest link qualities ($PRR < 0.2$) the combined metric loses their advantages.

1/x Combination

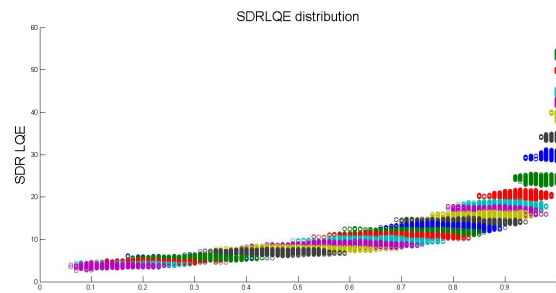
Looking at the correlation between *dSNR* and *CEPS* as displayed in Figure 6.8a we get an additional idea of combining the two metrics by division as the correlation shows a $1/x$ like behavior. This way we hope to extend the value range and clearly separate between different link qualities, as it was possible by linear combination.

$$SDRLQE_{div} = \frac{dSNR}{1 + m * CEPS}$$

The scaling parameter m is set the same way as for the linear combination to 4. How the metrics combined by division correlate to the basic link quality term *PRR* can be observed in Figure 6.8b. We can state that the extension of the value range is similar to the linear combination, but the spread of the values especially in the section above $PRR > 0.5$ is not as promising as the narrow distribution provided by the linear combination.



(a) dSNR CEPS correlation



(b) SDRLQE division combination

Figure 6.8: Combination of dSNR and CEPS by division

7

Comparison to Standard LQE

The next step in analyzing the possible contribution of software defined radios to link quality estimation is to compare the estimation performance of our novel metric estimators defined in Chapter 6. Through that comparison we hope to confirm our Hypothesis 5.6.1 stated at the end of Chapter 5. We start by choosing suitable competitive standard link quality estimators from the several types introduced in Chapter 2. In the second section we explain how we implemented the chosen estimators to appropriately work in our experiments. The different implementations will then be evaluated and compared on the different experimental scenarios. As a final experiment we left the lab setting of our artificially created data streams and let the common network performance measurement tool *iperf* generate traffic over our SDR setup. Additionally we run these experiments exposed to a periodic jammer to simulate a dynamically changing environment.

7.1 Choosing Competitive Standard LQE's

Based on the knowledge gained from the related literature studied in Chapter 2 the capabilities of GNU Radio and the characteristic of our experiments we choose the following set of standard link quality estimators.

- WMEWMA [10]
- ETX [27]
- RNP [28]
- Four-Bit [2]

Due to the fact, that the GNU Radio USRP2 platform does not provide signal power states directly, we could only choose LQEs which are based on packet statistics. But alternatively we use our digital SNR metric as a representa WMEWMA is LQE filtering PRR values to state the quality of a link. ETX, the expected transmission count is a receiver side metric widely distributed in wireless sensor networks and therefore an appropriate candidate for our comparison. Sender side candidates are RNP as a simple basic metric and Four-Bit which is the LQE actually implemented in ZigBee's contemporary Collection Tree Protocol (CTP) [5].

7.2 Implementing Standard Link Quality Estimators

Through our experiments on the five standard scenarios, we collected the necessary packet statistics to generate the chosen standard link quality estimators offline. Due to our complete logging of every sent and received packet, we know exactly which packets have been lost and can therefore recreate the given metrics.

7.2.1 WMEWMA

The first Link Quality estimator we choose is the WMEWMA as introduced by Woo and Culler in [10]. The value is generated in the form

$$WMEWMA = \alpha \times WMEWMA + (1 - \alpha) \times PRR.$$

The parameters were chosen to be as defined in [10] and used in a comparative simulation study by Baccour et. al. [31]. The filter coefficient α is therefore set to 0.6 and the new PRR fed into the filter is taken over a window of five packets.

How these values correlate in the model defining cable connection measurement we see in Figure 7.1. We see confirmed that the $WMEWMA$ value is directly a filtered estimate of the PRR value, therefore the computing for further comparison is quite straight forward.

$$PRR_{WMEWMA} = WMEWMA$$

7.2.2 ETX

Our second standard link quality estimator is ETX initially designed by Couto et. al. [27] for the DSDV and DSR routing protocols. ETX is computed at the receiver side the following way:

$$ETX = \frac{1}{PRR_{forward} \times PRR_{backward}}$$

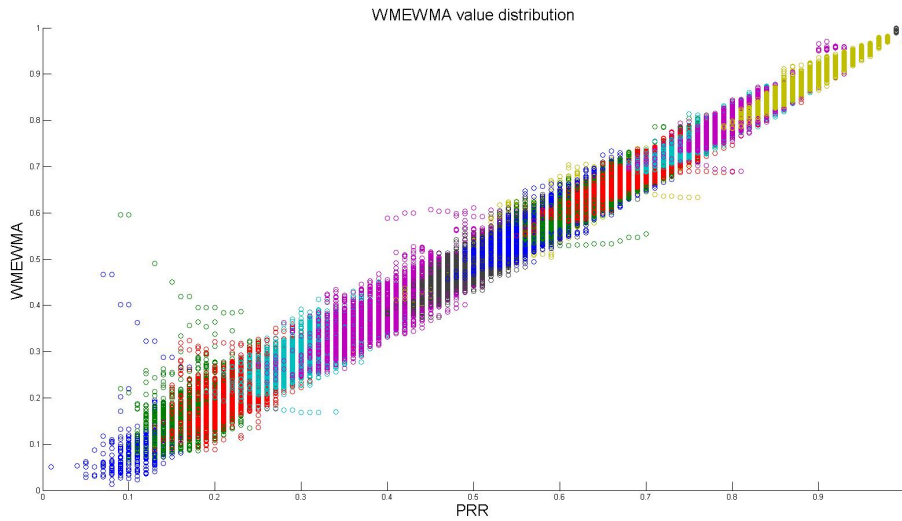


Figure 7.1: Correlation of the WMEWMA values to the actual PRR

So *ETX* is able to capture bidirectional issues not used in our lab experiment. Therefore we can set the Feedback parameter $PRR_{backward}$ to 1. By definition in [27] the current $PRR_{forward}$ is computed over a window of $w = 10$ packets.

Applying the estimator to our collected traffic data we get a correlation in the form shown in Figure 7.2 taken from the basic cable connection measurement. The characteristics of this correlation plot confirm the $1/x$ relation so to compare the precision of the estimator we take its inverse

$$PRR_{ETX} = \frac{1}{ETX}$$

7.2.3 RNP

RNP is the sender side counterpart of *ETX*. The slight difference is the generated value, as it counts retransmissions and not total transmissions, so a perfect channel has an *RNP* of 0 where the respective *ETX* value would be 1.

$$RNP = \frac{\text{Number of transmitted and retransmitted packets}}{\text{number of successfully received packets}} - 1$$

The limit and window over which a current *RNP* is determined is set to 10 to use the same range as for *ETX* and get useful short term measurements. This leads to the correlation presented in Figure 7.3.

This correlation looks already similar to the one computed for *ETX*. We later realized,

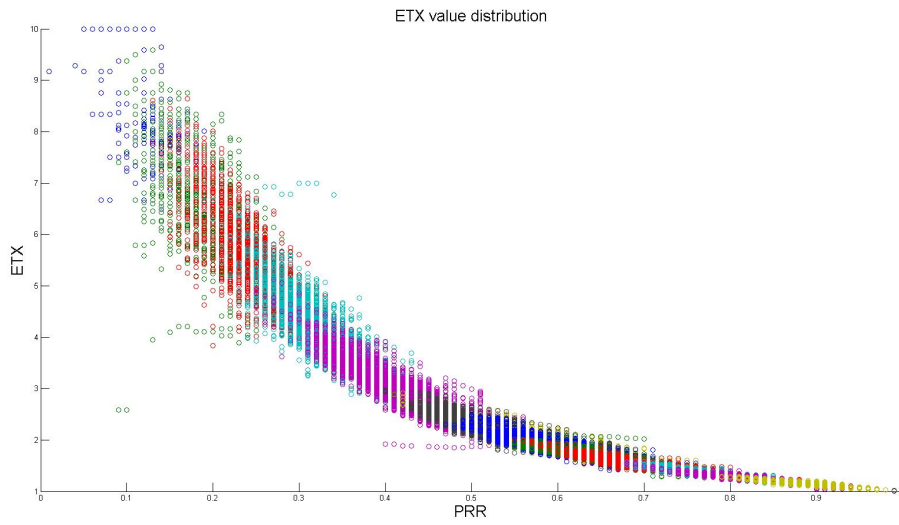


Figure 7.2: Correlation of ETX values to the actual PRR

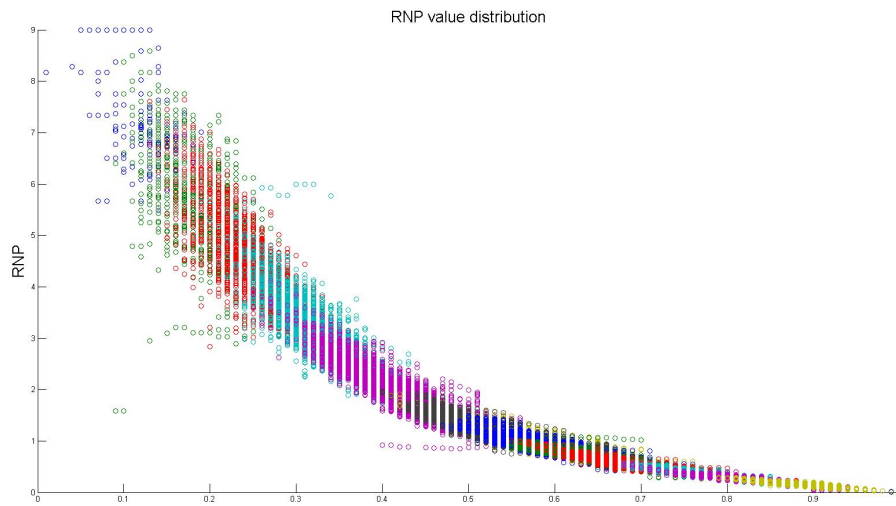


Figure 7.3: Correlation of RNP values to the actual PRR

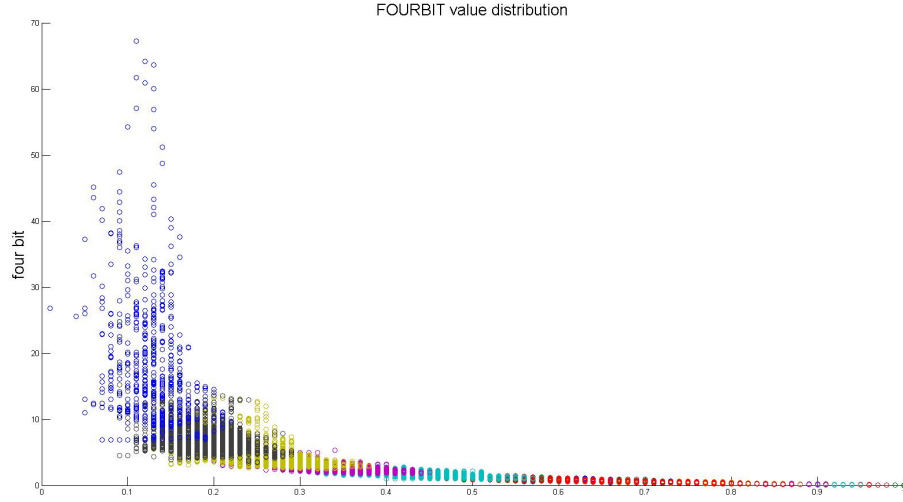


Figure 7.4: Correlation of Four-Bit values to the actual PRR

that after applying the model to estimate a PRR value based on RNP

$$PRR_{RNP} = \frac{1}{1 + RNP}$$

The resulting probability values were identical to the ones set by *ETX*. This is due to our experimental lab setup using a controlled one way traffic. So in performance analysis we will only consider *ETX* as reference value.

7.2.4 Four-Bit

The last standard link quality estimator we consider for our comparison is *Four – Bit*. *Four – Bit* generates an Estimation of *RNP* at the receiver side, which is the interesting one for us by filtering an estimated *ETX* value in the following way

$$estETX_{down} = \frac{1}{WMEWMA} - 1$$

Based on a window of $k_u = 5$ for unicast transmissions as in [2] the new $estETX_{down}$ values are computed and fed into the EWMA filter with filter coefficient $\alpha = 0.9$

$$FourBit = \alpha \times FourBit + (1 - \alpha) \times estETX_{down}$$

Through the double filtering the values are much more widespread as can be observed in the correlation in Figure 7.4

As *Four – Bit* is a filtered estimation of the RNP value, we calculate for comparison purpose the modeled PRR similar to the RNP approach.

$$PRR_{FourBit} = \frac{1}{1 + FourBit}$$

It has to be said that for modeled *PRR* estimates, we use sectionwise functions. So we will not allow *PRR* probability values less than 0 or greater than 1. Values jumping over the limits are therefore set to the corresponding limit values.

7.3 Performance Comparison of Link Quality Estimators

Having chosen and defined and adapted our competitive LQEs for our experimental setup, we now like to know how they perform in different scenarios over whole range of *PRR* link qualities. All of the estimators have their advantages in a certain application scenario but the basic work should still be to make a statement about which links are better than others. The more links you have available the more precise you should be able to differentiate between them. The LQE values are commonly used directly and stored for example in a routing table. To be able to compare we transformed the specific values into corresponding estimates of *PRR* link quality as this is directly linked to reliability and throughput. We first provide a detailed analysis of each scenario and continue to discuss the results in general.

7.3.1 Scenario Analysis

All six link quality estimators, the three defined by our SDR generated metrics and the three competing standard LQE which are based on packet statistics are assessed on the five basic scenarios of a cable connection, an indoor line of sight and a non line of sight measurement an outdoor run supplemented by a mobile scenario cover a wide range possible condition IEEE 802.15.4 transceivers have to cope with in real life applications. To reduce a certain statistical relevance we only considered *PRR* values with more than 10 occurrences. Which has proven to be a nice trade off between statistical relevance and capturing a sufficient number of packet for low quality channels.

Cable Connection

Damped by a 60dB attenuator to be able to simulate even the worst channel conditions, we first assess our estimator on the most static scenario. In Figure 7.5 we see how precise the estimates of the PRR where in comparison to the actual measured values. We observe that as expected , the dSNR estimator is the worst entering the transitional region but the absolute error never tops 13%. Best performing is the *WMEWMA* estimator which is not astonishing as the cable connection is a very stable scenario, so

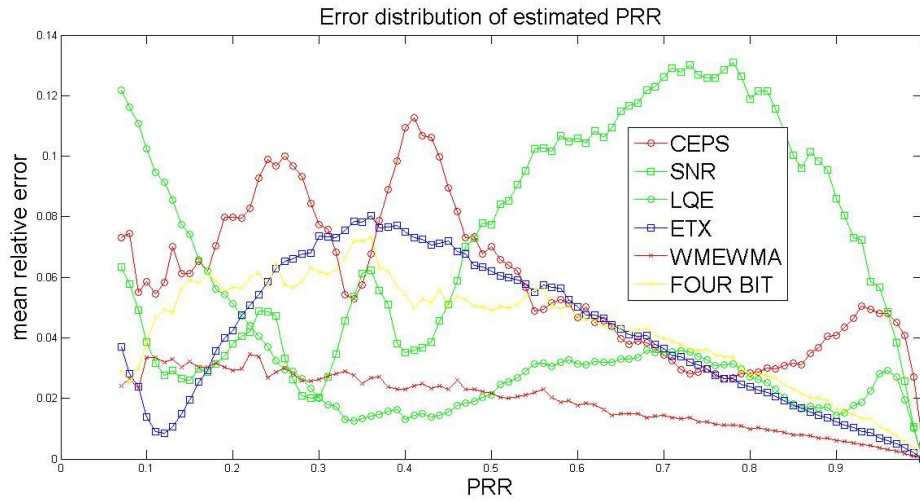


Figure 7.5: Error distribution of different LQE on a cable connection

the directly PRR based estimator is very precise $error < 4\%$. In general we see every estimator perfectly specifies the highest quality channels and gets worse for overall less channel quality. A general problem is that for low quality channels $PRR < 0.3$ we get less and less useful information as more than 70% of the packets are lost. Surprisingly the $CEPS$ based and especially the $SDRLQE$ estimators keep up with the established estimators despite of using only information gained from one packet.

Indoor Line of Sight

Replacing the cable connection with antennas, our first wireless scenario is an indoor line of sight measurements. In Figure 7.6 we detect the most significant changes to the cable connection scenario, that for the unshielded wireless connection the $dSNR$ gets almost useless and due to that distortion drags the $SDRLQE$ performance in similar scales. Astonishingly the *Four – Bit* estimator has some problems declaring good channels which could be due to the filtering of abrupt changes. Fortunately the $CEPS$ estimator performs except for some spikes below an estimation error of 10% and mostly close to the packed statistic based estimators.

Indoor Non Line of Sight

In the next scenario we add a metal cabinet into the line of site beam between the sender and the receiver to change the conditions of the experiment. We observe less distortion for the SNR therefore the $SDRLQE$ value is competitive again. The *Four – Bit* estimator performs normally again and the $CEPS$ estimator is only outperformed by the $WMEWMA$ LQE and keeps the absolute estimation error below 10%

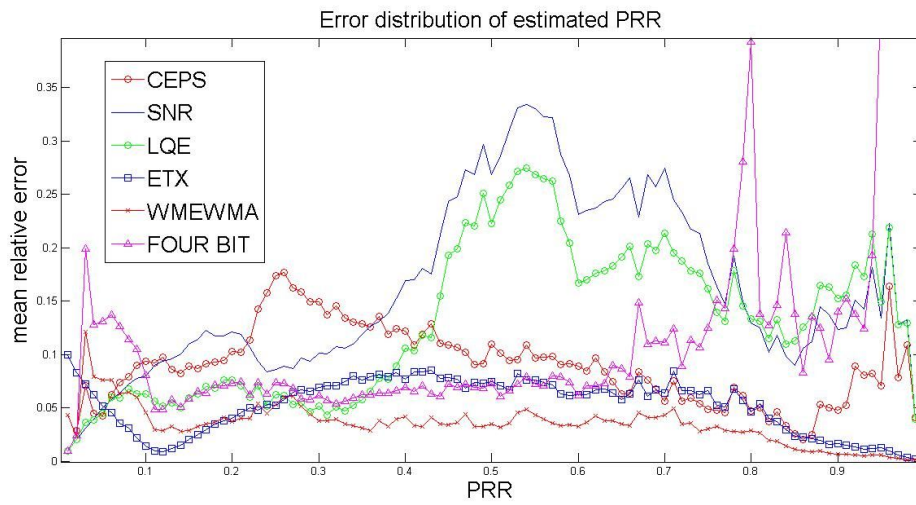


Figure 7.6: Error distribution of different LQE on a wireless LOS connection

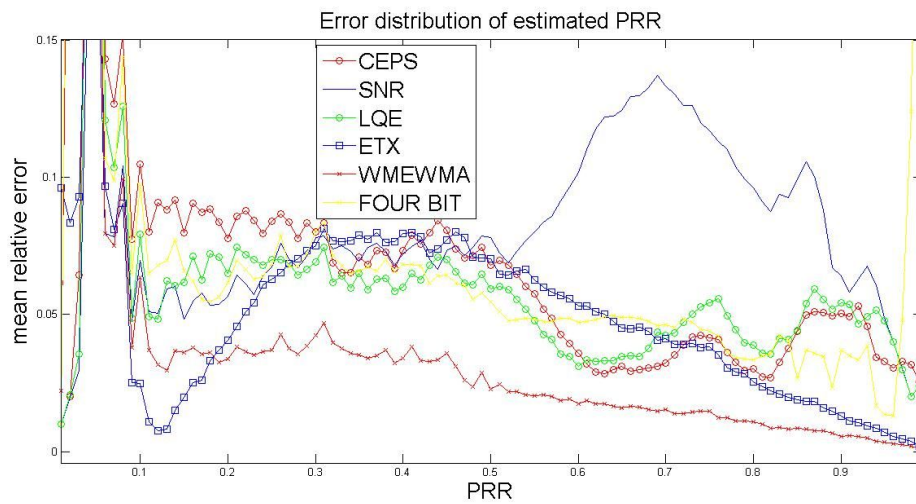


Figure 7.7: Error distribution of different LQE on a wireless NLOS connection

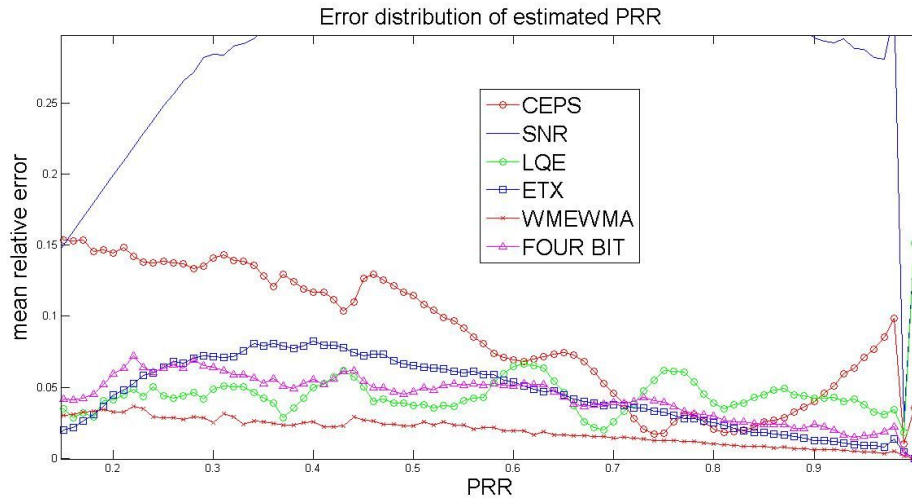


Figure 7.8: Error distribution of different LQE on an outdoor wireless LOS connection

Outdoor Line of Sight

The fourth experimental scenario we include in our analysis is an outdoor line of sight measurement. The $dSNR$ estimator shows in Figure 7.8 once more to be useless for the transitional region. For low quality channels The *CEPS* LQE is only half as precise as the packet statistic based LQEs but the *SDRLQE* estimator stays competitive. Overall the packet statistic based estimator perform well $error < 10\%$ over the whole range of *PRR* channel quality, which will be mostly due to the static conditions in the outdoor scenario with no changes in environment and almost no interfering radio signals from different sources.

Mobile Sender

The final standard scenario seems to be the most interesting one as it actively promotes non static channel behavior. Figure 7.9 presents us the facts that in general all LQEs perform worse. For the first time *WMEWMA* shows bigger problem for the fast changing low quality channels as well as the highly history dependent *FourBit* estimator which seems especially to have problems for channels resurrecting quickly from broken to perfect as it can happen in a mobile scenario including churn of sensors joining and leaving quickly. *CEPS* LQE displays similar performance as for the static scenarios despite some spikes for low quality links. The same is true for the *SDRLQE* estimator which suffers from $dSNR$ precision problems for high quality channels.

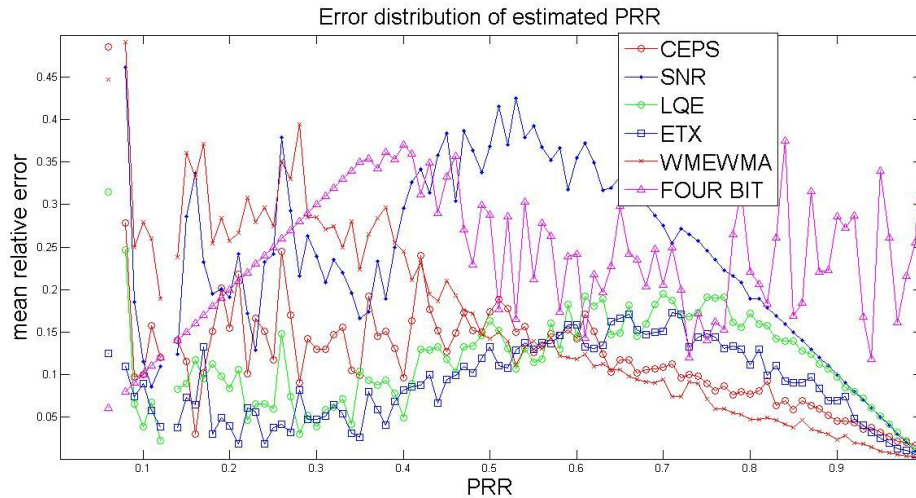


Figure 7.9: Error distribution of different LQE on a mobile wireless connection

7.3.2 Discussion of Precision and Timescale Performance

Analyzing the five scenarios has confirmed some of our assumptions and revealed some interesting facts. For the mostly static four scenarios the packet statistic based estimators or at least one of them were more precise than the *CEPS* based LQE. For the wireless scenarios, the well known problem of the *SNR* like metrics took effect and the only estimation possible was between highest quality channel and the rest. In one of four cases this dragged the *SDRLQE* estimator which is halfway dependent on the *dSNR* into imprecision. So in static scenarios *CEPS* and to some extent *SDRLQE* could provide an estimation precision of $\pm 10\%$. This in comparison to the best performing *WMEWMA* filter which reached a precision $\pm 5\%$ and better. Considering the timescale on which the information leading to the estimation the performance of the single packet based *CEPS* estimator is totally comparative as it uses at least 10 times less packets to collect the necessary information. The strength of the filter based *WMEWMA* and *Four – Bit* estimator storing the whole history of the channel in their old values in this static scenarios turns to a weakness in the mobile experiment where the channel changes often quickly, the knowledge of the history is not of much use. As the *CEPS* and *SDRLQE* estimators more or less keep their estimation performance the packet statistic based estimators perform significantly worse than for static scenarios. For low quality channels only *ETX* LQE reaches similar precision as for the static scenarios.

7.4 Network Performance Measurement Experiment

For our final experiment and the knowledge we gained analyzing the five environmental scenarios above, we designed a last set experiments closer towards an actual application

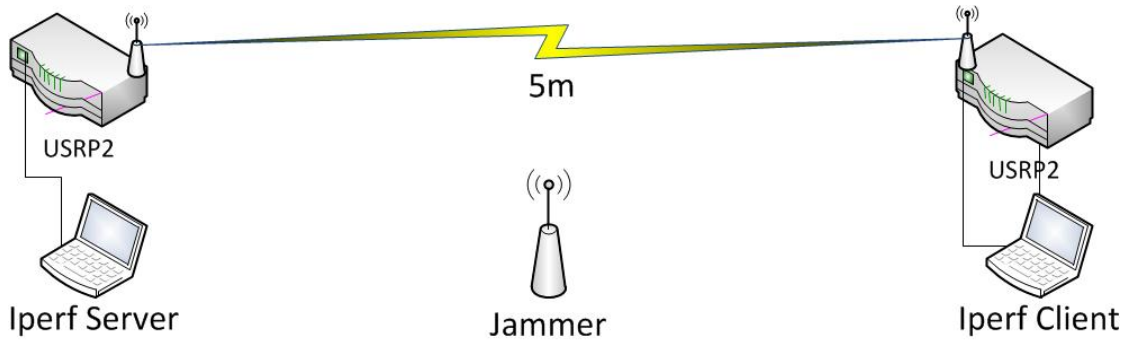


Figure 7.10: Iperf experimental setup

scenario. Up to now, the data traffic was clinically designed so we had the full knowledge of every chip sent and every chip received. We adapt the GNU Radio Code to directly generate the *CEPS* value at the receiver side and send externally generated UDP traffic over the adapted UCLA ZigBee [36] implementation. To artificially add dynamic behavior to the experiment, we add a jammer to the setup periodically changing the channel conditions in a more controlled way than experienced in the mobile scenario.

7.4.1 Integration of CEPS LQE into GNU Radio

Due to our complete picture of what was sent and what was received. We could compute the CEPS value directly comparing the received chip sequence to its sent counterpart. To integrate the CEPS value directly into the UCLA ZigBee Code we took the hamming distance to the best match symbol found in the symbol decoding method executed the *ieee802.15.4_packet_sink* signal processing block.

Algorithm 1 calculate CEPS

```

for  $i = 1 \rightarrow 16$  do
   $threshold = hammingdistance(chip, symbol(i))$ 
  if  $threshold < minthreshold$  then
     $bestmatch = symbol(i)$ 
     $minthreshold = threshold$ 
  end if
   $CEPS = minthreshold$ 
end for

```

7.4.2 Experimental Setup

Not to log every single chip frees some resources of the GNU Radio running notebooks. For the experiment, our UCLA Zigbee implementation provides the operating system a interface and a IP-address. This configuration allows us to run additional software

application on top of GNU Radio communication over the new interface using TCP/IP or UDP/IP communication. We are using iperf [47] in combination with jperf, a simple network performance measurement tool and its corresponding graphical user interface. Iperf analyzes the total amount of data transferred, throughput, jitter and important for us packet reception ratio, which is directly linked by packet size and measurement period to the throughput and total transferred data. We are sending 45Byte UDP packets which fit in a IEEE802.15.4 packet and result in 194 symbols sent over the channel. We are sending at a rate of 10Kbit/s as that can properly be sent received and processed by our configuration. For higher datarates above 18 Kbit/s we experience data loss due to reaching the limits of the computing power of the notebook pc's in service. The iperf sender and client are set 5m apart. The first set of experiments is run without jamming interference, the interferer experiments were run with a interferer sending on the same channel, but using a different higher bitrate and DSSS modulation so the the signal is not interpreted as data. The jammer periodically starts sending for approximately three seconds and then is silent for another three seconds. An overview of the setup placed in an indoor office environment is given in Figure 7.10.

7.4.3 Performance Evaluation

Every second the Iperf Server logs the amount of received data, bandwidth, totally transmitted and received packets including a packet loss rate, which is just the difference from packet reception rate to one. Unfortunately, one second is finest measurement unit Iperf provides. For our experimental configuration that means about 30 UDP packets are transmitted every second. To evaluate the precision performance of our link quality estimators we therefore take the opportunity and compare the estimated PRR values averaged over the very same packets analyzed by Iperf. We can exactly determine which packets to include in which estimation by interpreting the UDP sequence number number embedded in each packet we receive. Another important role the sequence number play for short term packet statistics necessary to generate *ETX*, *WMEWMA* and *Four – Bit* values. Besides precision evaluation it is at the same time interesting to observe the estimation values in time compared to the PRR values provided by the Iperf software tool especially for the jammed experiment runs. Figure 7.11 gives insight in such a time sequence of a jammed experiment run. Most affected by the jamming interferer is obviously the *dSNR* metric as shown in Figure 7.11a. Due to the drastically increased noise the *dSNR* estimates indicate nicely the jamming durations. We note further that the representative Iperf PRR values periodically change as well provoked by the jamming signal. In the same plot we get nicely displayed how the *CEPS* estimates behave similarly in comparison to the real measurement graph. Figure 7.11b presents us a corresponding section including the standard link quality estimators. We notice the the respectable precision even in the presented jamming case on the given second by second sampling, only the *Four – Bit* LQE is usually a bit late in adapting to changed conditions probably due to the high history control factor.

A summary of overall precision performance in several representa The far more in-

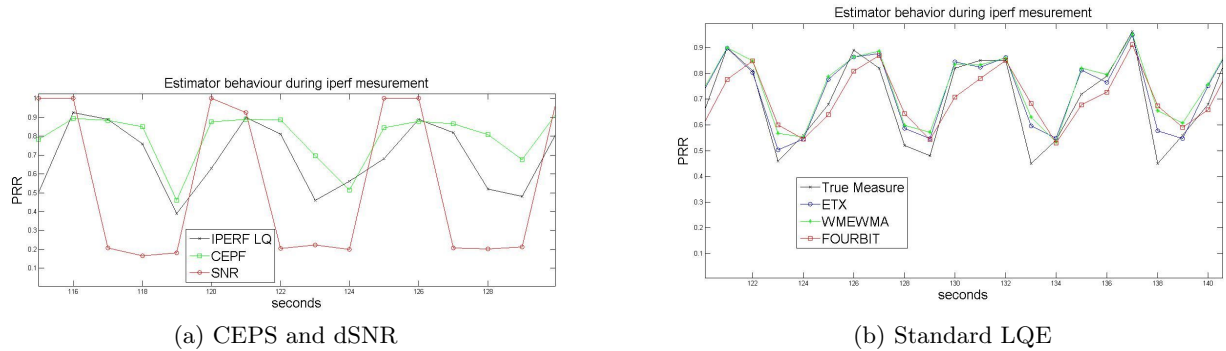


Figure 7.11: Estimator values in jammed timeseries

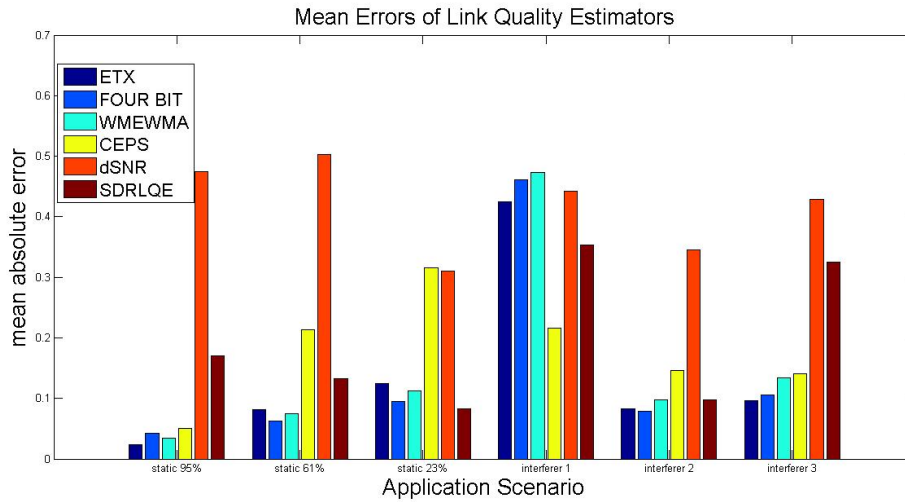


Figure 7.12: estimation precision on static and jammed iperf measurements

interesting results are gained from the interferer cases. The second and third jammed experiments show descent performance of *ETX*, *WMEWMA* and *Four – Bit* LQE followed by the only slightly less precise *CEPS* based estimator. *dSNR* shows in no run. *SDRLQE* at least in one of the jammed cases acceptable performance. The interesting outlier is the first jammed case. Special because it starts with longer burst of not received packets. This influence distorts the packet statistic based estimators so far, that they can not recover during the later run when packets are suddenly successfully received. *CEPS* LQE is the only Estimator showing at least a descent precision performance.

All these findings motivated us to run a final long term jammed experiment over 1000 seconds, smoothing certain short term characteristics. The results are presented in Figure 7.13 and confirmed earlier findings in different scenarios. The three packet statistic

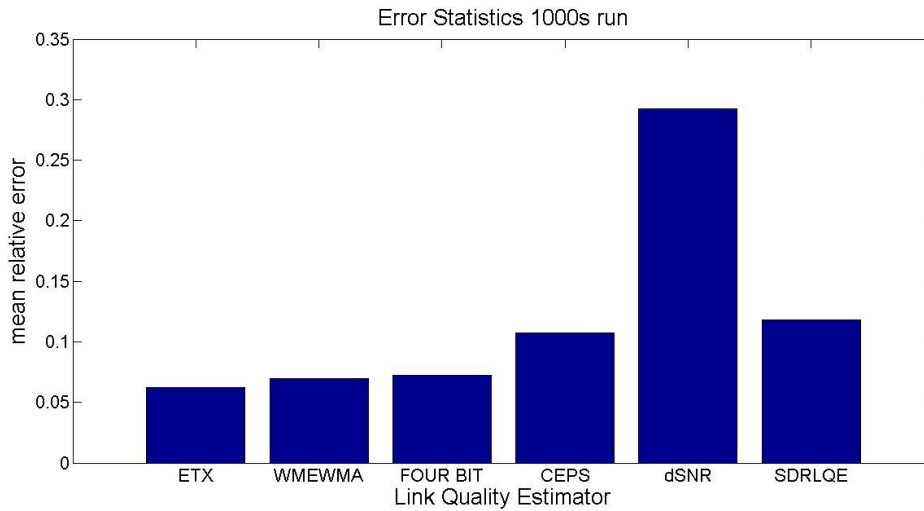


Figure 7.13: estimation precision in long term jammed iperf measurement

based estimators *ETX*, *WMEWMA* and *Four – Bit* deliver a precision slightly below $\pm 10\%$ while just behind follows *CEPS* reaching $\pm 12\%$ supplemented by an unreliable *dSNR* in a channel placed in the transitional region in this case around a PRR of 75% overall and an *SDRLQE* better but still too much dependent on useful *dSNR* values. So the *CEPS* metric harvested from only one packet, still provides us estimates reasonably close to estimates depending on a known history of 10 packets and more.

8

Conclusion and Outlook

8.1 Conclusion

Link quality estimation becomes more and more important as the frequency spectrum is limited and optimization approaches rely on a good knowledge about the conditions of the available wireless channels. We used recent advances in software defined radio to take a new approach in the area of link quality estimation. Our main contributions are:

- We analyzed the state of the art in link quality estimation and pointed out, that there is still a gap in time scale between link estimation based on received signal strength and packet statistic based estimation. Even some recent hybrid estimators [2, 48] try cross layer optimization but are still not capable to deal with condition changes as fast as one packet.
- The potential of software defined radios in terms of link quality estimation were explored. GNU Radio software and the UCLA ZigBee implementation of the IEEE 802.15.4 standard were used to show, that chip errors in a packet can be real world indicator to predict short term PRR
- We defined a new link quality estimator based on chip errors per symbol (CEPS) metric which proved to be more accurate than SNR based estimation especially in difficult channel conditions like mobile or jamming scenarios. At the same time CEPS based link quality estimation was still compatible to state of the art packet based link quality estimators despite of using only information of as little as the first two bytes of the packet payload.

8.2 Outlook

The introduction of software defined radios in the field of link quality estimation shows a promising new approach of gaining more reliable information about the channel conditions than received signal strength and packet statistics. Interesting next steps would be:

- A lot of packets in the transitional region get lost because they can not even synchronize properly. In the case of IEEE 802.15.4 this means no synchronization zero is found. A first goal would be to adapt the demodulation process in a way to be able to collect more information of heavily broken packets. A problem is that this could be a waste of energy by always scanning the channel for broken packets when we do not even know if something was sent.
- The use of software defined radio bears the potential of other indicators able to inform upper layers about the quality of the wireless channels. A recent paper of Qin et. al. [49] introduces error vector magnitude and a spectrum factor (SF) but conclude as well, that no one of these single metrics is sufficient for different channel conditions and qualities
- GNU Radio is in continuous progress. A next step would be to integrate ZigBee like applications on top of the IEEE 802.15.4 standard to determine the influence of different newly found link quality estimators on the application performance and even further in specialized ad hoc networks like mesh networks, vehicular networks, sensor networks and opportunistic networks.
- Hybrid link quality estimators like Fourbit [2] already use cross layer optimization techniques to improve routing performance in protocols like the Collection Tree Protocol (CTP) [5] or the more agile Backpressure Collection Protocol (BCP) [50]. BCP for example calculates link states packet by packet and could therefore improve further by including current link quality information gained from CEPS measurements.
- Another area of research where cross layer optimization of plays an important role is the field of cognitive radio where even cross platform optimization is a topic. So integrating the new link quality metrics into a cognitive engine of such a multi-protocol capable device could offer a lot of benefit to optimize future wireless communication.



CD-ROM Content

Below you find a list of the files provided alongside the Thesis. The Folder *adapted ZigBee versions* includes the adapted source code of the UCLA ZigBee implementation of [36]. It embodies the sender and receiver side of the initial experiments and the version of the final experiments assisted by the iperf application.

In the *GNU Radio 3.2.2 Source* Folder you find the openly available GNU Radio software platform used in this thesis. The *LaTeX Source* folder provides you with the necessary source files of the report.

Matlab was used to analyze and extract information out of the logfiles generated by the software defined radio implementation. In the *Matlab Analysis Files* folder you find on the one hand the Matlab scripts used to process the logfiles on the other hand you are provided with pre processed scenario data used to generate all the plots starting from Chapter 6.

The *References* folder holds all the cited documentation and referenced papers.

Finally you find useful guidelines of installation and execution for the provided set of platforms and applications in the HowTo file and the of course the final report in digital form.

It follows the content of the CD-ROM provided in addition to the document:

```

- adapted ZigBee Versions
  |-- gr_quadrature_demod adaptions
  |-- ZigBee_iperf
  |-- ZigBee_Receiver
  |-- ZigBee_Sender
- GNU Radio 3.2.2 Source
  |-- gnuradio-3.2.2.zip
- LaTeX Source
  |-- Thesis_Tex_Source.zip
- Matlab Analysis Files
  |-- scenario data
  |
  |-- analyze_app_errors.m
  |-- analyze_chip_errors.m
  |-- analyze_symbol_errors.m
  |-- analyze_sync.m
  |-- assign_rssi.m
  |-- assign_rssi_app.m
  |-- beta_factor.m
  |-- beta_scatter.m
  |-- change_msb.m
  |-- Check_CRC.m
  |-- chip_errormask.m
  |-- compare_chip_sequence.m
  |-- error_dist_plot_los.m
  |-- error_distribution_detailcable.m
  |-- error_distribution_los.m
  |-- error_distribution_mob.m
  |-- error_distribution_nlos.m
  |-- error_distribution_od.m
  |-- find_missing_packets.m
  |-- find_missing_packets_app.m
  |-- frame_plot.m
  |-- iperf_plot.m
  |-- local_analysis.m
  |-- local_analysis_iperf.m
  |-- local_analysis_size.m
  |-- local_analysis_synch.m
  |-- metric_fairness.m
  |-- miss_placeholder.m
  |-- number_of_symbol_plot.m
  |-- oQPSK_MSK_table.m
  |-- quickplot_cable_etx.m
  |-- rearrange_packets.m
  |-- rearrange_packets_app.m
  |-- remove_missing_packets.m
  |-- satur.m
  |-- scenario_plot.m
  |-- scenario_plot_longsync.m
  |-- scenario_plot_los.m
  |-- scenario_plot_mob.m
  |-- scenario_plot_od.m
  |-- scenario_plot_size.m
  |-- scenario_plot_sync.m
- References
  |-- basic references
  |-- additional references
  |-- documentations
  |-- Task_Description.pdf
- HowTo.txt
- Master Thesis.pdf

```

B

Abbreviations

| | |
|---------|---|
| ADC | Analog-Digital Converter |
| CDMA | Code Devision Multiple Access |
| CD-ROM | Compact Disc Read-Only Memory |
| CPU | Central Processing Unit |
| CRC | Cyclic Redundancy Check |
| CSMA-CA | Carrier Sense Multiple Access - Collision Avoidance |
| dB | Decibel |
| DSSS | Direct Sequence Spread Spectrum |
| FCF | Frame Control Field |
| FCS | Frame Check Sequence |
| GNU | GNU's Not Unix! |
| IEEE | Institute of Electrical and Electronics Engineers |
| kb/s | kilo bits per second |
| LOS | Line of Sight |
| LQ | Link Quality |
| LQE | Link Quality Estimator |
| LR-WPAN | Low-Rate Wireless Personal Area Network |
| MAC | Media Access Control |
| MFR | MAC Footer |
| MHz | Mega-Hertz |
| MPDU | MAC Protocol Data Unit |
| MSK | Minimum-Shift Keying |
| NLOS | None Line of Sight |
| O-QPSK | Offset Quadrature Phase-Shift Keying |
| OSI | Open Systems Interconnection |
| PAN | Personal Area Network |
| PC | Personal Computer |
| PER | Packet Error Rate |
| PHR | PHY Header |
| PHY | Physical Layer |
| PN | Pseudo-Random Noise |
| PPDU | PHY Protocol Data Unit |
| PRR | Packet Reception Rate |
| RNP | Required Number of Packet Retransmissions |
| QPSK | Quadrature Phase-Shift Keying |
| SDR | Software Defined Radio |
| SF | Spreading Factor |
| SFD | Start-of-Frame Delimiter |
| SHR | Synchronization Header |
| SNR | Signal to Noise Ratio |
| USRP | Universal Software Radio Peripheral |
| WMEWMA | Window Mean Exponentially Weighed Moving Average |
| WLAN | Wireless Local Area Network |
| WPAN | Wireless Personal Area Network |

Bibliography

- [1] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, ser. MobiSys '04. New York, NY, USA: ACM, 2004, pp. 125–138. [Online]. Available: <http://doi.acm.org/10.1145/990064.990081>
- [2] N. Baccour, A. Koubaa, H. Youssef, M. Ben Jamaa, D. do Rosario, M. Alves, and L. Becker, "F-lqe: A fuzzy link quality estimator for wireless sensor networks," in *Wireless Sensor Networks*, ser. Lecture Notes in Computer Science, J. Silva, B. Krishnamachari, and F. Boavida, Eds. Springer Berlin / Heidelberg, 2010, vol. 5970, pp. 240–255.
- [3] T. Schmid, "Gnu radio 802.15. 4 en-and decoding," UCLA NESL, Tech. Rep., 2005.
- [4] I. D. C. (IDC). (2011, February) Framingham. [Online]. Available: <http://www.idc.com/about/viewpressrelease.jsp?containerId=prUS22689111§ionId=null&elementId=null&pageType=SYNOPSIS>
- [5] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis, "Four-bit wireless link estimation," 2008.
- [6] R. Fonseca, S. Ratnasamy, J. Zhao, C. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point routing in wireless sensor networks," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 2005, pp. 329–342.
- [7] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 14–27. [Online]. Available: <http://doi.acm.org/10.1145/958491.958494>
- [8] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 1–14. [Online]. Available: <http://doi.acm.org/10.1145/1644038.1644040>

- [9] K. Srinivasan, M. A. Kazandjieva, S. Agarwal, and P. Levis, “The beta-factor: measuring wireless link burstiness,” in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 29–42. [Online]. Available: <http://doi.acm.org/10.1145/1460412.1460416>
- [10] K. Srinivasan and P. Levis, “Rssi is under appreciated,” in *Proceedings of the third workshop on embedded networked sensors (EmNets)*, vol. 2006. Citeseer, 2006.
- [11] J. Mitola *et al.*, “Cognitive radio: an integrated agent architecture for software defined radio,” *Doctor of Technology, Royal Inst. Technol.(KTH), Stockholm, Sweden*, pp. 271–350, 2000.
- [12] D. E. Albert Cerpa, Naim Busek, “Scale: A tool for simple connectivity assessment in lossy environments,” Center for Embedded Networked Sensing, University of California, Los Angeles (UCLA), Tech. Rep., 2003.
- [13] A. Woo, Alec and A. Culler, David, “Evaluation of efficient link reliability estimators for low-power wireless networks,” 2003. [Online]. Available: <http://techreports.lib.berkeley.edu/accessPages/CSD-03-1270.html>
- [14] D. LaI, A. Manjeshwar, F. Herrmann, E. Uysal-Biyikoglu, and A. Keshavarzian, “Measurement and characterization of link quality metrics in energy constrained wireless sensor networks,” in *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, vol. 1, dec. 2003, pp. 446 – 452 Vol.1.
- [15] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks,” in *Proceedings of the 1st international conference on Embedded networked sensor systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 1–13. [Online]. Available: <http://doi.acm.org/10.1145/958491.958493>
- [16] M. Zuniga and B. Krishnamachari, “Analyzing the transitional region in low power wireless links,” in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, oct. 2004, pp. 517 – 526.
- [17] L. Tang, K.-C. Wang, Y. Huang, and F. Gu, “Channel characterization and link quality assessment of iee 802.15.4-compliant radio for factory environments,” *Industrial Informatics, IEEE Transactions on*, vol. 3, no. 2, pp. 99 –110, may 2007.
- [18] M. Z. n. Zamalloa and B. Krishnamachari, “An analysis of unreliability and asymmetry in low-power wireless links,” *ACM Trans. Sen. Netw.*, vol. 3, June 2007. [Online]. Available: <http://doi.acm.org/10.1145/1240226.1240227>
- [19] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, “Link-level measurements from an 802.11b mesh network,” in *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, ser. SIGCOMM '04. New York, NY, USA: ACM, 2004, pp. 121–132. [Online]. Available: <http://doi.acm.org/10.1145/1015467.1015482>

- [20] D. Kotz, C. Newport, and C. Elliott, “The mistaken axioms of wireless-network research,” Citeseer, Tech. Rep., 2003.
- [21] D. De Couto, D. Aguayo, B. Chambers, and R. Morris, “Performance of multi-hop wireless networks: Shortest path is not enough,” *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 83–88, 2003.
- [22] R. Kave, R. Khalili, and K. Salamatian, “Evaluation of packet error rate in wireless networks,” 2004.
- [23] R. Khalili and K. Salamatian, “A new analytic approach to evaluation of packet error rate in wireless networks,” 2005.
- [24] Chipcon, *CC2420 - 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*, 2009. [Online]. Available: www.chipcon.com
- [25] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, “Complex behavior at scale: An experimental study of low-power wireless sensor networks,” Citeseer, Tech. Rep., 2002.
- [26] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, “Understanding the causes of packet delivery success and failure in dense wireless sensor networks,” in *Proceedings of the 4th international conference on Embedded networked sensor systems*, ser. SenSys '06. New York, NY, USA: ACM, 2006, pp. 419–420. [Online]. Available: <http://doi.acm.org/10.1145/1182807.1182885>
- [27] Z. Jian and Z. Hai, “A link quality evaluation model in wireless sensor networks,” *Sensor Technologies and Applications, International Conference on*, vol. 0, pp. 1–5, 2009.
- [28] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, “Predictable 802.11 packet delivery from wireless channel measurements,” *SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 159–170, August 2010. [Online]. Available: <http://doi.acm.org/10.1145/1851275.1851203>
- [29] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, “a high-throughput path metric for multi-hop wireless routing,” *Wireless Networks*, vol. 11, pp. 419–434, 2005. [Online]. Available: <http://dx.doi.org/10.1007/s11276-005-1766-z>
- [30] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin, “Temporal properties of low power wireless links: modeling and implications on multi-hop routing,” in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '05. New York, NY, USA: ACM, 2005, pp. 414–425. [Online]. Available: <http://doi.acm.org/10.1145/1062689.1062741>
- [31] C. Boano, M. Zuniga, T. Voigt, A. Willig, and K. Romer, “The triangle metric: Fast link quality estimation for mobile wireless sensor networks,” in *Computer*

- Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on*, aug. 2010, pp. 1–7.
- [32] N. Baccour, A. Koubaa, M. Ben Jamaa, H. Youssef, M. Zuniga, and M. Alves, “A comparative simulation study of link quality estimators in wireless sensor networks,” in *Modeling, Analysis Simulation of Computer and Telecommunication Systems, 2009. MASCOTS '09. IEEE International Symposium on*, sept. 2009, pp. 1–10.
- [33] *IEEE Standard for Information technology - Telecommunications and information exchange between systems - 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 Computer Society Std.
- [34] Z. Alliance, “Zigbee specification,” *ZigBee document 053474r06, version*, vol. 1, 2005.
- [35] “Adaptive home control system.” [Online]. Available: <http://www.adhoco.com/>
- [36] T. S. D. R. Inc., “What is software defined radio?” [Online]. Available: www.sdrforum.org
- [37] L. Choong, “Multi-channel ieee 802.15.4 packet capture using software defined radio,” 2009. [Online]. Available: <http://nesl.ee.ucla.edu/document/show/297>
- [38] B. E. et. al., “Gnu radio.” [Online]. Available: www.gnuradio.org
- [39] M. Ettus, “Ettus research llc.” [Online]. Available: www.ettus.com
- [40] J. Notor, A. Caviglia, and G. Levy, “Cmos rfc architectures for ieee 802.15. 4 networks,” *Cadence Design Systems, Inc*, 2003.
- [41] T. M. Inc., “Matlab r2010b,” Software, 2010. [Online]. Available: www.mathworks.com
- [42] B. Han, L. Ji, S. Lee, B. Bhattacharjee, and R. Miller, “All bits are not equal - a study of ieee 802.11 communication bit errors,” in *INFOCOM 2009, IEEE*, april 2009, pp. 1602–1610.
- [43] K. Wu, H. Tan, H. Ngan, Y. Liu, and L. M. Ni, “Chip error pattern analysis in ieee 802.15.4,” *IEEE Transactions on Mobile Computing*, vol. 99, no. PrePrints, 2011.
- [44] K. Wu, H. Tan, H.-L. Ngan, Y. Liu, and L. Ni, “Measurement study of mobility-induced losses in ieee 802.15.4,” in *Communications (ICC), 2010 IEEE International Conference on*, may 2010, pp. 1–5.
- [45] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 95–107. [Online]. Available: <http://doi.acm.org/10.1145/1031495.1031508>

-
- [46] E. W. Weisstein, “Sigmoid function,” *From MathWorld—A Wolfram Web Resource*, 2011. [Online]. Available: <http://mathworld.wolfram.com/SigmoidFunction.html>
- [47] NLANDR/DAST, “iperf.” [Online]. Available: <http://sourceforge.net/projects/iperf/>
- [48] M. H. Alizai, O. Landsiedel, J. A. B. Link, S. Götz, and K. Wehrle, “Bursty traffic over bursty links,” in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 71–84. [Online]. Available: <http://doi.acm.org/10.1145/1644038.1644046>
- [49] Y. Qin, Z. He, and T. Voigt, “Towards accurate and agile link quality estimation in wireless sensor networks.”
- [50] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, “Routing without routes: the backpressure collection protocol,” in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, ser. IPSN 10. New York, NY, USA: ACM, 2010, pp. 279–290. [Online]. Available: <http://doi.acm.org/10.1145/1791212.1791246>

