If You Can't Take The Heat: Temperature Effects On Low-Power Wireless Networks And How To Mitigate Them

Florian Schmidt[1], Matteo Ceriotti[2], Niklas Hauser[1], and Klaus Wehrle[1]

[1] Communication and Distrib. Systems Group, RWTH Aachen University, Germany
{schmidt,wehrle}@comsys.rwth-aachen.de, niklas.hauser@rwth-aachen.de
[2] Networked Embedded Systems Group, University of Duisburg–Essen, Germany
matteo.ceriotti@uni-due.de

Abstract. Low-power wireless networks, especially in outdoor deployments, are exposed to a wide range of temperatures. The detrimental effect of high temperatures on communication quality is well known. In this paper, we use a testbed with self-made temperature control devices to investigate the effects of temperature on several communication-relevant metrics. The analyses both confirm some previously published results and demonstrate deviations from others. Based on these results, we propose a Reed–Solomon-based FEC scheme to mitigate the negative effects of temperature and provide results suggesting that such a scheme is both feasible and advantageous.

Keywords: wireless sensor networks; measurements; packet corruption; bit errors; reliability; forward error correction; temperature effects

1 Introduction

Low-power wireless networked devices are seeing more and more uses, enabling monitoring of areas both remote and inaccessible, and within our own homes, from any point on Earth. Depending on the deployment scenario, those devices can be exposed to strongly varying environmental effects. In recent publications, it has been shown that temperature has a strong effect on communication quality [2–5]: as temperature rises, communication becomes more challenging, up to an eventual complete breakdown. To further investigate these effects, we designed "HotBox", a solution to exactly control temperature and spatial orientation of sensor nodes. While our experiments confirm the influence of temperature, they also highlight some deviations. In particular, we observe a more marked link quality decrease in the case of a heated receiver, in contrast to the more significant impact of a heated transmitter demonstrated in the literature.

Based on the gathered measurements, we propose a solution to offset the effect that temperature has on the wireless communication by using an adaptive Reed–Solomon-based FEC mechanism, and discuss preliminary results. This paper presents our investigation and paves the way for both gathering further knowledge about the interplay between temperature and low-power wireless communication, as well as possible counteractions to preserve system reliability.

2 Related Work

We first turn our attention to the current knowledge about the impact of temperature on the reliability of low-power wireless communication. Then, we survey FEC schemes proposed for low-power wireless networks.

Bannister et al. [2] were one of the first to analyze the impact of temperature on the performance of the CC2420 radio, the one employed in our study. The results demonstrated a reduction of RSSI with an increase of temperature, more marked with a heated transmitter. In [3], this behavior was also identified, again with larger differences when the transmitter was heated than when the receiver was. Both studies identify the cause in the loss of gain in the CC2420 Low Noise Amplifier. The asymmetry was confirmed by Boano et al. [4] for Noise Floor, PLR and LQI in a comprehensive study on the effects of temperature. Their TempLab [5] setup is based on remotely-controlled IR light bulbs, polystyrene foam enclosures, and Peltier elements to build cheap and small temperature chambers. In our work, we develop a similar solution to experiment with temperature and spatial orientation of devices, which we make available for everybody to reproduce [9]. Using this testbed, we extend the available knowledge by demonstrating deviations from the previously reported behavior, in particular showing a greater impact of a heated receiver on the decrease in link quality.

FEC increases transmission reliability by recovering corrupted payloads by using correction information added to transmitted messages. The additional per-message overhead is justified by the reduced number of retransmissions required for a successfully delivery, reducing overall energy consumption. Reed–Solomon (RS) codes [7], a popular choice for FEC, can be very efficient in correcting corrupted messages while maintaining energy efficiency. Moreover, changing the code rate dynamically [1] at run time based on link quality outperforms static RS codes. Finally, Hermans et al. [6] exploit knowledge about mutation patterns caused by the CC2420's MSK demodulator used to receive OQPSK modulated signals. They build probability distributions to infer the most likely transmitted symbol and reconstruct the original data. In our work, we propose an adaptive FEC scheme that exploits the knowledge about error distributions inside corrupted packets and the impact of temperature on the reception probability.

3 Influence of Temperature on Communication

In the following, we present results from our experimental setup investigating the influence of temperature on the communication quality between sensor nodes.

3.1 Experimental Setup

For our experiments, we used TelosB sensor nodes from different manufacturers as well as production runs and ages. The TelosB is a widely-used platform that employs a CC2420 radio chip for communication which implements the IEEE 802.15.4 standard. At the physical layer, the standard defines a DSSS OQPSK modulation in the 2.4 GHz ISM band, with a nominal data rate of 250 kbps. All experiments were conducted in a room that witnesses little interference from surrounding IEEE 802.11 (WiFi) networks; furthermore, we used channel 26 of the 802.15.4 standard, which is outside the band allocated to 802.11 in Europe.

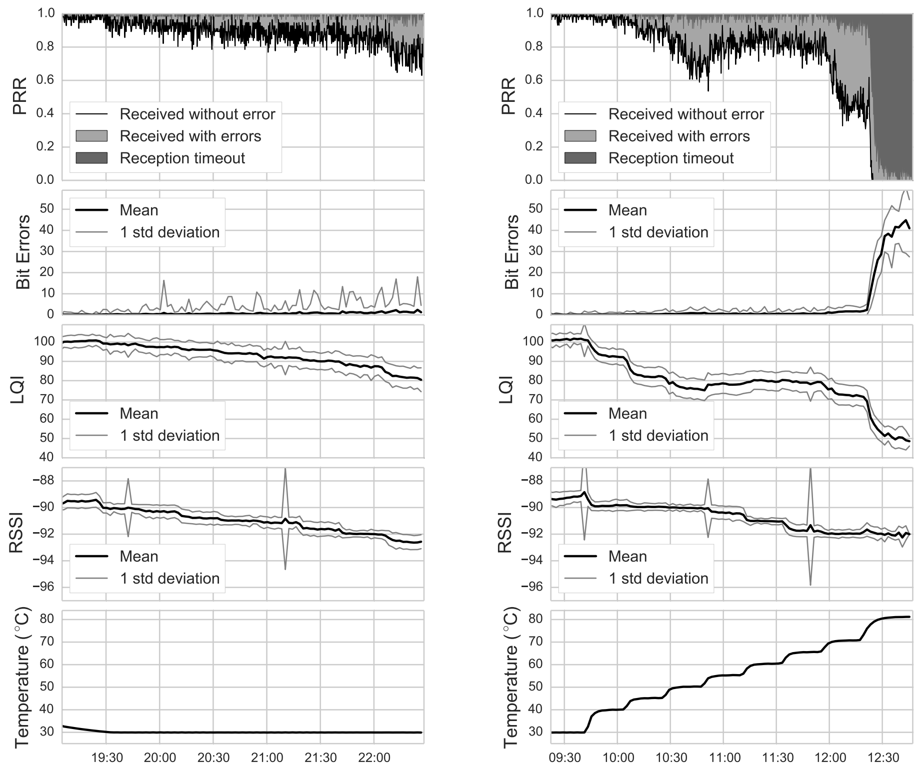
For temperature control during the experiments, we designed a system we termed "HotBox". Its design stems from the need to accurately control the influence of temperature on sensor motes. One of our goals with HotBox was to design a highly accurate control system (less than 0.5 °C deviation) which can be produced relatively quickly and cheaply. Thus, all hardware elements are off-the-shelf items, while all manufacturing can be done with a soldering iron, a PCB mill, and a laser cutter, which are often available via the rapidly-spreading FabLab concept. For reasons of brevity, we refrain from an in-depth description and performance evaluation of HotBox and refer to [9] for further information.

Experiments used direct (single-hop) connections between links. Each experiment comprised two boxes with one node each. Both nodes were connected to a PC via USB; the PC created the packets and sent them via USB to one node (alternating the sender role between the two nodes every packet); the node would then send the packet via the CC2420 radio. If the other node received the packet, it forwarded the received version to the PC via USB, which then compared the original and the received version for bit errors. Otherwise, a timeout would be triggered at the PC to identify the missed reception. Each run comprised 180 000 packets, spread out over approximately 3.5 hours, during which we gradually increased the temperature in one of the boxes from 30 °C to 80 °C, while keeping the other at 30 °C. We exchanged the nodes between experimental runs to account for potential performance differences between production runs and different models of the TelosB nodes. However, for the metrics presented in this paper, we could not find any noticeable performance differences.

3.2 Packet Error Dependency on Temperature

First, we aimed to reproduce the results from previous work [2–5] in our setup. We started with a temperature of 30 °C and gradually increased the temperature in 5 or 10 °C increments, spending 20 minutes at each target temperature. Results for each node were saved separately, thereby creating two separate datasets from heating the transmitter and from heating the receiver, respectively. We recorded the RSSI, the Link Quality Indicator (LQI), and bit error rate (BER) as well as packet reception rate (PRR). We split the latter into three cases: a packet could be received without errors; received, but with errors; or not received at all (completely lost). Figure 1 shows a typical result from one of our experiments. All presented results are values as witnessed by the receiving mote. Thus, the temperature shown in Figure 1b shows temperature changes throughout the experiment. Conversely, Figure 1a does not show any changes in the temperature, because it was the transmitter which was heated, while the receiver, whose values are shown, was kept at a constant temperature.

Overall, it can be seen that all metrics are negatively influenced by temperature. However, the amount as to which they are influenced differs, and heat-



(a) Transmitter heated

(b) Receiver heated

Fig. 1: Influence of temperature on several key communication metrics. Note that all results are collected at the receiver, hence temperature in Fig. 1a stays stable because only the transmitter is heated. While temperature has a negative effect on all metrics, the effect is generally much stronger if the receiver is heated.

ing the transmitter and the receiver has different magnitudes of effect for most metrics. The only metric that is largely independent of this fact is the RSSI. All other metrics show a much higher negative influence when the receiver is heated. Heating the transmitter to 80 °C still allows communication, albeit with a packet error rate of more than 20%. In contrast, communication completely breaks if that temperature is applied at the receiver's side, and even at 70 °C, PER is much higher at above 50%. This is reflected in the BER, which explodes at receiver temperature above 70 °C. At the same time, LQI significantly decreases.

Summarizing, our results reinforce the notion of temperature as a significant influence on the communication quality in low-power networks. However, we were not able to reproduce the results in [4], which showed transmitter heating as the larger influence on quality metrics. In our experiments, heating the receiver produces a larger impact. This has repercussions that we will discuss later.

4 Temperature-Based FEC for Sensor Nodes

These results show that, while bit error rate increases with temperature, it typically does not do so massively until the point the connection collapses completely. However, even those relatively small increases already lead to steadily and significantly increasing packet error rates. Considering these results, we investigated the use of an FEC scheme that introduces redundancy into the sent messages in challenging link conditions. Such a system should (1) be adaptable so that an optimum tradeoff between reliability and overhead can be chosen, and (2) be computationally simple enough to work well on constrained devices. These requirements suggest the use of Reed–Solomon (RS) codes. Moreover, readily available implementations for constrained devices already exist, e.g., TinyRS [7].

A Reed–Solomon code [8] is parameterized with a tuple (m, n, k), where m is the size of a block in bits, and the code transforms k blocks into n blocks with $n > k$ (which gives a so-called *code rate* of k/n), being able to correct up to $\lfloor (n - k + 1)/2 \rfloor$ erroneous blocks (that is, blocks with at least one flipped bit) in the resulting message. We decided to use 8-bit blocks because byte-level operations are efficient to use on microcontrollers.

4.1 Simulator

To investigate the effects of RS-based FEC on reception quality, we need to exactly reproduce the environmental effects during each experiment. Otherwise, the differences in channel quality impair the comparability of results in different runs, because the effects of channel conditions mix with those of different FEC strengths. However, repeatability of wireless testbed results is a well-known hard problem. To abstract from channel conditions, we created a simple trace-based simulator. In such a simulator, a trace (i.e., a recording of a real-world communication that contains bit errors, environmental conditions, etc.) is used to translate bit errors that occurred in the real world onto a simulated connection.

The simplest way to use such a trace would be to mark which bits were corrupted, and overlay this pattern 1:1 onto another message, regardless of length and contents of both messages. This, however, would discount the differences in relative errors depending on message content. It has been shown [6, 10] that different nibbles (4-bit blocks) of data have different error rates. Hence, we first extracted from the trace the relative BER for each nibble. The simulator would then compute 16 values, one for each nibble, and normalize them to an average of 0 (resulting in some negative and some positive values). During the experiment, whenever a packet was prepared for sending by the PC, instead of forwarding it to the mote via USB, we handed it to the simulator. The simulator took the next packet in the trace, counted its bit errors and calculated the BER for that packet. It then added the relative per-nibble values to that rate, and finally applied, for each nibble in the simulated message, the corresponding per-nibble BER to its 4 bits. The resulting (potentially corrupted) packet was then handed back to the PC application, which would compare it to the original version.

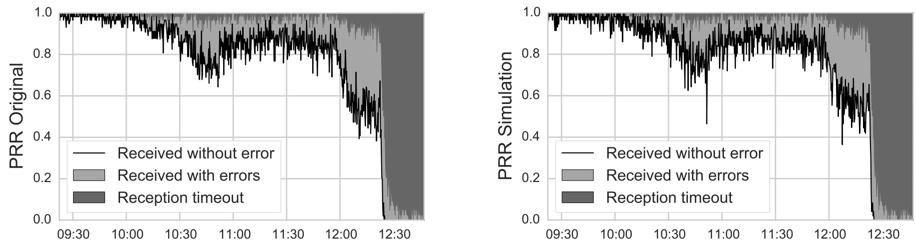


Fig. 2: Comparison of packet reception rates between the real-world experiment presented in Figure 1b and a simulation run based on that experiment's trace. The simulator closely follows the real-world results.

Figure 2 shows a comparison of a real-world measurement and the trace-based simulation of random packet contents of the same size. The simulation models the real-world results well; it only slightly overestimates packet error and loss rates. This means that results produced with the simulator will potentially underestimate the efficacy of our FEC scheme, but not overestimate it.

4.2 Evaluation Setup

To minimize the differences between the packets in the trace and simulated packets, we required all packets to have the same size. For our experiments, we used 80 bytes of payload. However, different robustnesses of RS codes mean that for a fixed k, n has to be increased. Due to the fixed packet size, we decided to do the opposite: for the baseline experiment without any FEC, we sent 80-byte payloads. As robustness increased, code length n stayed the same, but data length k was reduced accordingly. Thus, at a code rate of $1/2$, the packet length was still 80 byte, but it only carried 40 bytes of data, plus 40 bytes of redundancy.

Under these circumstances, packet reception rate is not a meaningful metric any more. Normally, if the amount of data is kept static, the packet size increases with stronger FEC to accommodate the additional code bytes. This serves as a trade-off in itself, since as the packets grow in size, the chance of having more bit errors within a packet increases, too. If packet size is kept static, packet reception rate will only increase with increasing FEC robustness, as more and more errors can be repaired. We therefore account for the amount of data bytes in each packet by calculating a *normalized throughput* metric: $T(80, k) = \frac{k}{80} \cdot \frac{PRR_{decoded}}{PRR_{received}}$, where k is the number of data bytes in the 80-byte payload, and $PRR_{received}$ and $PRR_{decoded}$ are the rate of packet with errors before and after Reed–Solomon error correction, respectively. Thus, T yields a value of 1 for an unencoded connection without packet losses. As robustness increases (and therefore the amount of data bytes in the 80-byte packet decreases), $k/80$ decreases: the potential maximum throughput is reduced due to coding. The code can offset this by repairing corrupted packets and therefore increasing $PRR_{decoded}/PRR_{received}$.

4.3 Evaluation Results

We then used traces from the experiments described in Section 3 to drive our simulator. To investigate the effects of different Reed–Solomon code robustnesses, we repeated the experiments with different settings of data length k. The results presented in Figure 3 are based on the trace of the measurement shown in Figure 1b; hence, the packet reception rates follow the same general behavior.

The figure clearly shows that the performance of FEC strongly depends on the channel conditions. At low temperatures, channel conditions are unproblematic, so the unencoded connection shows the highest throughput. As temperature rises, however, FEC shows its advantages: while each message can transport less information, the information is more robust and more rarely lost. At the very high end, when communication breaks down almost completely, higher and higher code rates are needed to keep at least some messages uncorrupted.

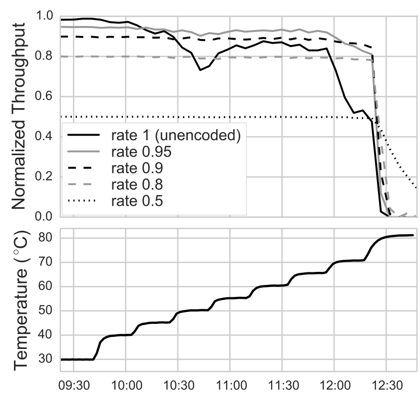


Fig. 3: Reed–Solomon FEC increases effective throughput in challenging conditions by significantly reducing packet loss. This simulation is based on the results presented in Figure 1b. Even using as little as 5% of the message for FEC (rate 0.95) produces a large benefit as soon as temperatures rise above 40 °C.

Interestingly, even under conditions that are common and not especially challenging (40 °C are easily reached in watertight containers under sunshine), FEC can already provide noticeably fewer packet losses with a small overhead, keeping communication both more stable and reliable, and often also increasing the throughput. We therefore suggest to always consider whether adding FEC will provide beneficial effects to your low-power wireless connectivity, especially if challenging conditions cannot be ruled out or can even be expected at least intermittently.

5 Discussion and Future Work

We originally devised the presented FEC scheme in the hope that we could reproduce previous results [4], which showed that the sender's temperature has a larger effect on communication quality than the receiver's. This would allow for a temperature-driven adaptive FEC, where the robustness of the Reed–Solomon code is increased as temperature at the sender rises. That decision would stem from purely locally available information: the temperature of the mote itself.

However, we could not reproduce this effect in our experiments. Instead, the receiver's temperature always had a larger influence on quality metrics such as packet reception rate or LQI than the sender's temperature. Therefore, the temperature that produces the larger effect is not available locally to the sender, and cannot be used to decide on code rate before sending of the message.

We envision several possibilities that warrant further scrutiny. An adaptive FEC scheme could nevertheless solely rely on sender temperature, which does have some impact on the communication; such an adaptation would therefore be merely suboptimal. When mote temperatures correlate strongly, for example, inside rooms or on flat terrain, purely local information may already be sufficient. Alternatively, a system of feedback should also be investigated. For example, motes could inform their neighborhood about their temperature by piggybacking this information onto other messages. However, if we already consider feedback, we should not only focus on temperature. In the end, high communication performance always depends on low packet loss rates. Instead of feeding back information about an influence factor on packet loss (temperature), information about packet loss rate itself could be fed back from the receiver to the sender. If the communication already uses acknowledgments, this information can be effectively inferred by the sender "for free" from the number of received acknowledgments. It can then be used to adapt the code rate to minimize packet losses. At this point, such an FEC scheme closely resembles rate adaptation schemes as found in WiFi networks. We consider the investigation whether and in what fashion the myriad of contributions in the field of WiFi rate adaptation can be applied to low-power wireless networks an exciting field for future work.

Acknowledgments This research was partially funded by the Alexander von Humboldt Foundation.

References

1. Ahn, J.S., Hong, S.W., Heidemann, J.: An adaptive fec code control algorithm for mobile wireless sensor networks. J. Comm. Netw. 7(4), 489–498 (Dec 2005)
2. Bannister, K., Giorgetti, G., Gupta, S.K.S.: Wireless sensor networking for "hot" applications: Effects of temperature on signal strength, data collection and localization. In: Proc. HotEmNets. ACM (2008)
3. Boano, C.A., et al.: The impact of temperature on outdoor industrial sensornet applications. IEEE Trans. Ind. Informat. 6(3), 451–459 (Aug 2010)
4. Boano, C.A., et al.: Hot Packets: A systematic evaluation of the effect of temperature on low power wireless transceivers. In: Proc. ExtremeCom. ACM (2013)
5. Boano, C.A., et al.: Templab: A testbed infrastructure to study the impact of temperature on wireless sensor networks. In: Proc. IPSN. IEEE Press (2014)
6. Hermans, F., et al.: All is not lost: Understanding and exploiting packet corruption in outdoor sensor networks. In: Proc. EWSN, pp. 116–132. Springer (2014)
7. Liang, C.J.M., Priyantha, N.B., Liu, J., Terzis, A.: Surviving Wi-fi Interference in Low Power ZigBee Networks. In: Proc. SenSys. pp. 309–322. ACM (2010)
8. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics 8(2), 300–304 (Jun 1960)
9. Schmidt, F., Ceriotti, M., Hauser, N., Wehrle, K.: Hotbox: Testing temperature effects in sensor networks. Tech. Rep. AIB-2014-14, RWTH Aachen (2014)
10. Schmidt, F., Ceriotti, M., Wehrle, K.: Bit error distribution and mutation patterns of corrupted packets in low-power wireless networks. In: ACM WiNTECH (2013)