Wireless Sensor Networks
To Hop or Not to Hop

David Jensen

Stud. M.Sc. CE, ASE, Aarhus University
11229@post.au.dk

Henrik Bagger Jensen

Stud. M.Sc. CE, ASE, Aarhus University
201304157@post.au.dk

Christian M. Lillelund

Stud. M.Sc. CE, ASE, Aarhus University
201408354@post.au.dk

Troels Kvistgaard Thomsen

Stud. M.Sc. ET, ASE, Aarhus University
09641@post.au.dk

May 29, 2018

Abstract

Direct or relayed communication between nodes in a wireless sensor network is one of the corner stones of electrical and computer engineering. The life time and reliability of wireless nodes often becomes the bottle neck of pioneering ideas, so the usage of multiple nodes to sustain efficiency can be a necessity. Typically, when a consumer node cannot reach a wireless router in a wireless local area network (WLAN), other fixed access points must relay the packets of the consumer. In a wireless sensor network (WSN), other concerns also come into play, such as network dynamicity, battery levels and noise interference, which must be considered before relaying packets. In this project a relaying protocol was build based on signal strength and accumulated transmission errors (lost packets) The protocol tries to predict when to change between direct communication and relaying, while the project tries to validate the quality of the designed and implemented protocol for the different conditions of transmission range and channels. The protocol quality is determined in terms of the event quality (how many times did we notice an event), energy consumption aiming to use the least amount of energy to extend lifetime, or conceivably a compromise of both. Events in our case is the heart rate of an athlete runner running an oval-formed running track. We find that creating such a multivariate decision protocol is indeed possible and results show that the effect on event quality overshadow the drawback of intensified energy consumption when increasing the length of the running track.

I. Introduction

This report details a project done in Wireless Sensor Networks (WSN) with a mini-project called "To hop or not to hop". The project follows the original idea of determining when to relay packets in a wireless network, but with a little twist to it - instead of placing the receiving node in a fixed position, we place it on an athlete running a marathon on a oval-formed running track in order to measure his heart rate every minute (events). This information is going to be transfered to a base station over a wireless connection with an added two relay stations in between that help ensure sufficient network coverage of the track. To design the track for our purpose, we will employ knowledge of fading, radio wave propagation and received input power (dbM) in a wireless node.

We will seek to create an effective protocol that can determine when to communicate directly with the node carried on the athlete/runner and when it is best to use one of the relays. This protocol will take transmissions errors and signal strength into consideration. For inter-node connectivity, we will design and implement the data-link layer stop-and-wait ARQ method on all nodes.[1]. The protocol is used to answer the question if a WSN network should primarily relay packets based on event quality in

[1] [Chipcon Products()]

order to pick up the most events or go for the option that requires the least amount of energy to extend the lifetime of the network.

With reference to the mini-project presentation, we use telosb nodes all running TinyOS with a packet size of 128 bytes. The RF transceiver is a single-chip 2.4 GHz IEEE 802.15.4 compliant CC2420 with a data rate of 250 kbps.

Section II details the theory behind the implementation in section III. Set in a real-life scenario, we performed a test of the WSN with results in section VI. The conclusion of our project is in section VIII.

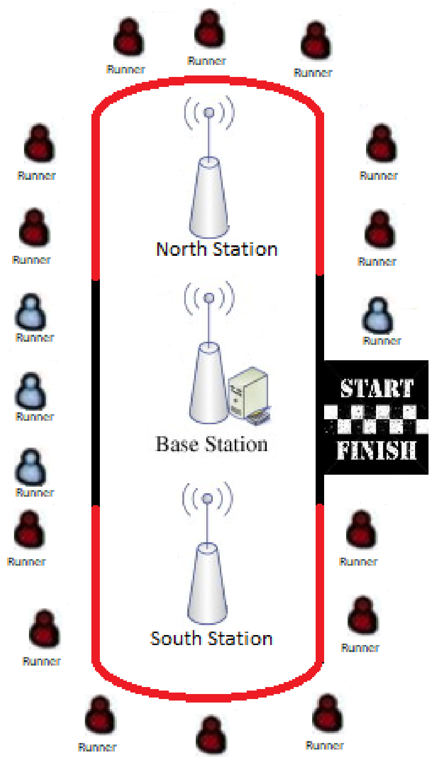

Figure 1: *A base station requests heart rate information from a marathon runner racing around a track. In the red northern territory, the runner is out of range, so the base station transmits requests to the north station which relays the message and likewise with the south station.*

II. Theory

This section introduces theory concepts used in this project. We will cover both antenna theory, go into detail about fading and explain how the ARQ method works.

i. Path loss: "Free space vs. building"

To be able to understand the need for relaying, one must understand the boundaries of the chosen working environment. The range equation gives a good theoretical reference point to how a far a certain quality signal can be transmitted in optimal conditions. The range equation is dependent on transmission frequency and the characteristics of a transmitting and receiving antenna. The frequency dependency comes from the range equation calculation of the far field distance from the antenna pair. This far field distance occurs when the magnetic and electric part of the signal has a steady state phase relation. Also, the distance is dependent on the size and type of the antenna, while the telosb has a 2.7cm inverted f-antenna transmitting at a 2.408GHz center frequency. A 2.408GHz transmission frequency is chosen from the standard IEEE_802.15.4, making 2.401GHz a "1" and 2.408 a "0". Trying to keep the frequency as low as possible gives longer transmission ranges both in theory and in practice, as the lower frequencies have better penetration chances given obstacles. The radiation pattern, $-3dB$ power line, of a Ferrite-based inverted F-antenna can be seen in figure 2 [2]. Focusing on protocol and power consumption, the range equation will be visualized based on an omni-directional antenna with same polarization, that is isotropic.

The telosb software specifies transmission power in dBm and the receiving/transmitting antenna have same the characteristics, so the need to understand the antennas workings beyond the far field distance estimation is not needed for this project. Since our main focus is a packet control protocol, each antenna is treated as isotropic being able to broadcast up to 100m in each direction as specified in the data sheet of the CC2420[3]. To simulate a real scenario, the transmitting antenna could be mounted on a rota-

[2] [Ka'bi(2016)]
[3] [Chipcon Products()]

tional motor following the runner using computer vision hence the main beam, e.g. 153 deg, of the antenna pattern would always point towards the runner verifying our calculation approach. Giving a far field distance of $0.01167m$ and optimal conditions in air, figure 4 shows the expected received signal strength indication in dBm based on the range equation 3 and the assumed assumptions.

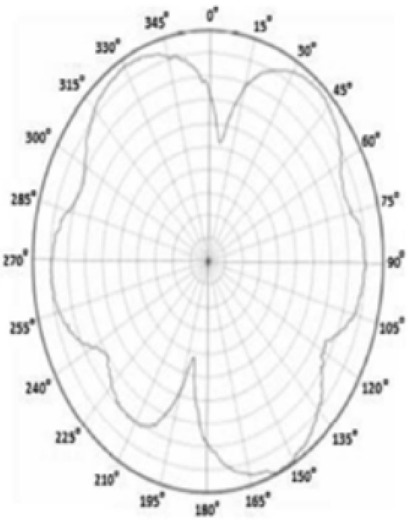

Figure 2: *Estimation of an inverted-F antenna's radiation pattern.*

$$f = 2.407 \cdot 10^9 Hz \qquad D = 2.7 \cdot 10^{-3} m$$
$$\lambda = \frac{c}{f} = 0.125m \qquad \gamma_{air} = 2 \qquad \gamma_{building} = 5.5$$
$$d0 = \frac{2 \cdot D^2}{\lambda} = 1.171 \cdot 10^{-4} m$$

$$P_{rcvd}(d) = \frac{P_{tx} \cdot G_t \cdot G_r \cdot \lambda^2}{(4\pi)^2 \cdot d^2 \cdot L} \tag{1a}$$
$$= \frac{P_{tx} \cdot G_t \cdot G_r \cdot \lambda^2}{(4\pi)^2 \cdot d_0^2 \cdot L} \cdot \left(\frac{d_0}{d}\right)^2 \tag{1b}$$
$$= P_{rcvd}(d_0) \cdot \left(\frac{d_0}{d}\right)^{\gamma} \tag{1c}$$

Figure 3: *Range equation based on a 2.7cm wide inverted f-antenna and a 2.408GHz center frequency*

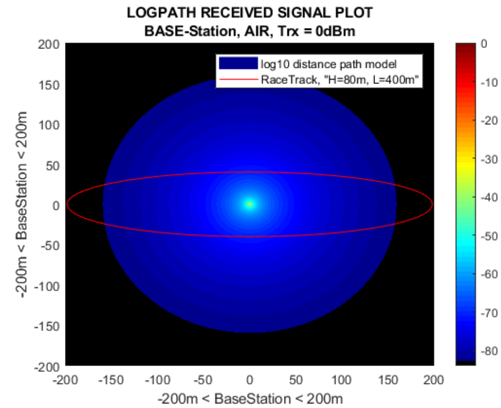

Figure 4: *$-84dBm$ $159.3m$ received signal area of the running track. The antenna cannot cover the entire track.*

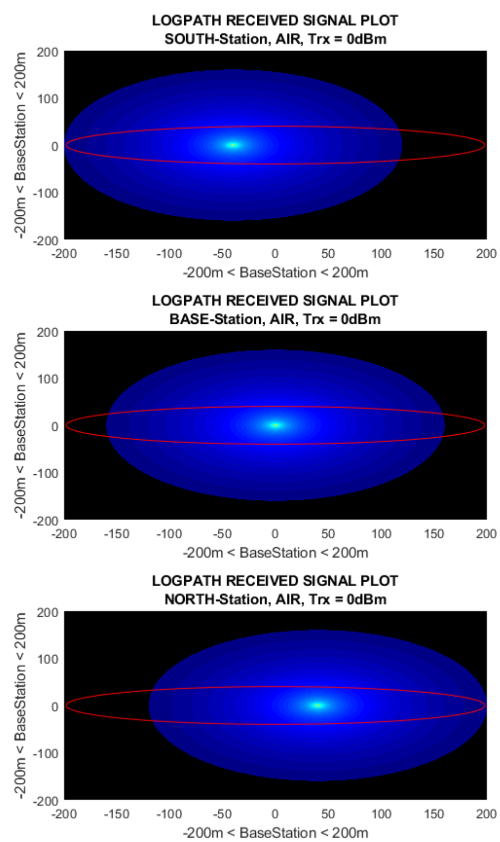

Figure 5: *Three stations covering the entire track individually.*

Given a racetrack of 400m in width, under optimal conditions as shown in figure 4 a telosb node will not be able to cover the entire track and two additional relay "hop" stations must be installed to provide sufficient coverage. The antenna dimensions and transmission power gives a natural boundary to which relaying will be the only option. Figure 5 shows the RF input power of two relay stations in comparison to the base station, while figure 4 6 shows the combined RF input power of the runner node.

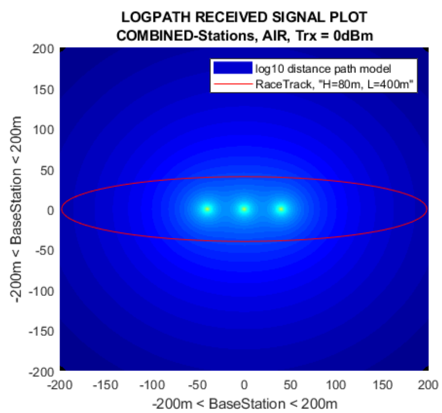

Figure 6: *Three stations covering the entire track combined.*

Lowering the transmission power can prolong the lifetime of individual nodes and the wireless network all together, but at the cost of less coverage. Figure 7 shows the single and 8 the combined RF input power of the runner node with all nodes using transmission power of $-24dBm$. The size of the track is now only 7.5% of the full power track.

Further reduction in RF input power can happen for multiple reasons, e.g: reflection, diffraction, scattering and Doppler fading. An easy noise model can be to change γ_{air} in equation 3 to $\gamma_{building}$ with values taken from the WSN book[4]. Figure 9 and 10 show the distance at minimum transmitted power inside a building providing a stunning 0.035% of the coverage related to full power in open AIR.

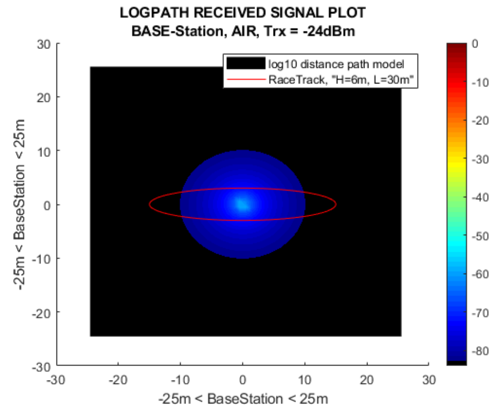

Figure 7: 10.1m *adequate RF input power for the base station.*

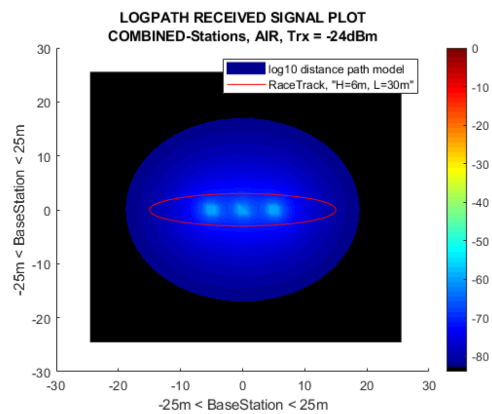

Figure 8: 10.1m *Three stations combined RF input power.*

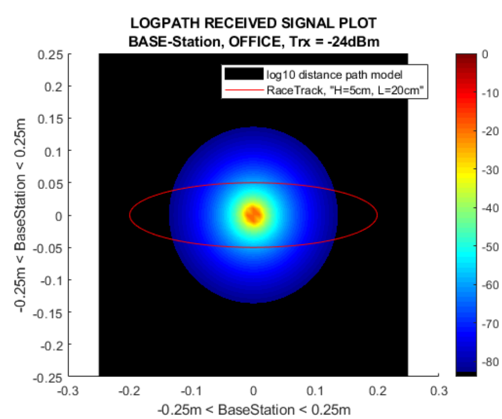

Figure 9: 13.1cm *adequate RF input power for the base station.*

[4][Karl and Willig(2006), p. 98]

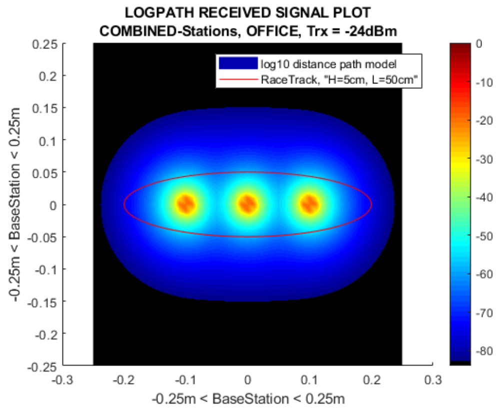

Figure 10: $13.1cm$ *Three stations combined RF input power.*

ii. Fading

The above plots all give a good estimate of the RF input power in a stationary clean environment with fixed-positioned nodes, but in case of a dynamic environment with obstacles, aspects like fading must be considered. Intrinsic and extrinsic electronic noise impacts the signal to noise ratio (SNR) floor of the received signal at a receiving node, but more expensive electronics can compensate for induced noise. If no mobile phones are at the track, it can reduce the noise even further, however one would still experience RF input power drops at times in a dynamic environment.

The broadcasting behavior of the WSN causes constructive and destructive interference at the receiver, which can lead to deep fading. The phenomenon can occur when two or more, out of phase, adequate signals arrive at the receiver simultaneously leading to a critical destructive interference. Deep fading causes the signal strength to fall below the established noise floor, deeming the signal unreliable or unmeasurable. If a receiver has experienced a deep fade, depending on the time length or number of lost packages, it can either be categorized as a fast fade or a slow fade. More on these terms later. The fading is typically caused by reflection, diffraction or scattering of the signal, causing in line of sight (LOS) signal interfering with a none line of sight (NLOS) signals. When nodes move relative to each other it can not only cause interference, but also a change in behavior. An example of this is the Doppler fading, in which the signal tends to shift in frequency relative to the movement of the source and the node.

ii.1 Doppler fading

Following the standard IEEE_802.15.4, it permits the telosb node to transmit in the ISM band at frequencies between 2.4 and $2.4835GHz$. Having 12 channels, $2MHz$ wide and separated by $5MHz$, the telosb allows a center frequency signal to shift $\pm 1MHz$ while still being acknowledged by the receiving channel. If the signal shift the frequency more than $1MHz$, the receiving node will simply filter out the signal and the packet will be lost. The sign of the shift in frequency varies if the distance between the motes is increasing or decreasing. We will have a changing position of the runner relative to the base station at different speeds due to the track shape, so an investigation of the effects was made.
The distance between a packet, every quarter of a second, covered by a runner, running at $12kph$, was calculated closest and furthest on the track relative to the base station. The runner is running circular, but the change in distance experienced by the base station will by a straight line leading to Doppler frequency found at different speeds relative to track position. The results showed a $29.698Hz$ frequency shift at the end of the track, while at the top of the track a $26.773Hz$ shift happened. Since the telosb has a buffer of $1MHz$, the results put our misgivings about Doppler fading to rest. For the sake of scalability and flexibility, a extreme case was also calculated for the system: Had the runner been running at approximately $50000kph$, Doppler fading would have been an issue. Given the calculated results, the project is solely focusing on fast and slow fading as simulated instead. See appendix 1 for Doppler calculations.

ii.2 Fast fading and slow fading

Busty bit errors at the receiver is often measured in clusters with different duration or length. Fast fading clusters are typically in the range of tens to hundreds of milliseconds, before the received signal again is adequate, while slow fading clusters are in the range of tens of seconds to minutes. Fast fading and slow fading are both by-products of the broadcasting behavior of the nodes. While no clean separation can be made between the two, slow fading is referred to as a shadowing effect and fast fading can be simplified to reflection, e.g. a signal at $2.4GHz$ will have a wavelength of $12.5cm$, given an opposite phased signal every $12.5cm$. If the $2.4GHz$ LOS signal travels $1m$ to the node and the NLOS signal travels $1.125m$ to the node, they would cancel each other out.

The calculations of the reflected fading also must consider the directivity of the antenna, since the signal strength also varies at different angles from the source. An antenna with high directivity will not have a full cancellation at the node, since the LOS signal will have a higher amplitude than the NLOS signal. The drop-in signal strength can be estimated to around $30-40dB$ $(60-70dBm)$[5] and these are the values for both slow and fast fading chosen for the simulations in this project. Slow fading can represent diffraction and scattering of the signal from objects in between source and node, e.g. more runners on the track or a photographer taking pictures along the route. The slow fading is modeled as a random stochastic variable with a showing variance visualized through a log-normal fading plot. Given a shadowing variance of $2.22dBm$ and probability of occurrence at 10% for both fast and slow fading, fast fading effect is simulated to last $333ms$ while slow fading is simulated to last $14s$. Figure 11 plots the base station's RF input power including fading and 14 plots the binary, received or lost package, output of the base station. Figure 12 plots the RF input power of the stations combined including fading and 15 plots the binary output of the stations combined. Figure 13 and 16 show a plot of combined stations, but with all three station signals being victims of individual fading patterns.

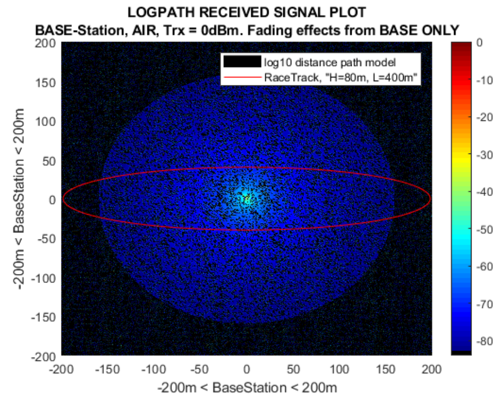

Figure 11: *Base station RF input power plot after it has experienced fading.*

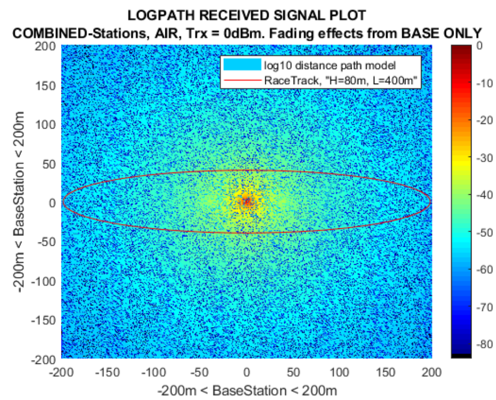

Figure 12: *Combined stations RF input power plot after base station experienced fading.*

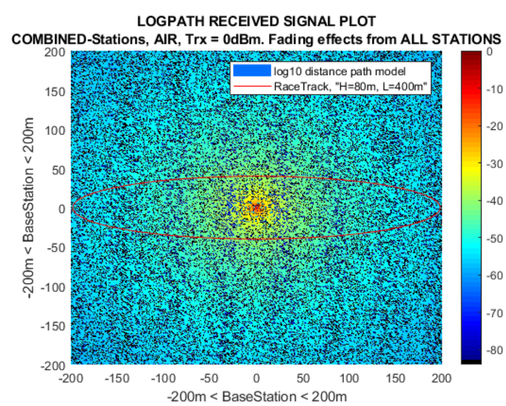

Figure 13: *Combined stations RF input power plot after all stations experienced fading.*

[5] Appendix: 1, [Karl and Willig(2006)] page 92

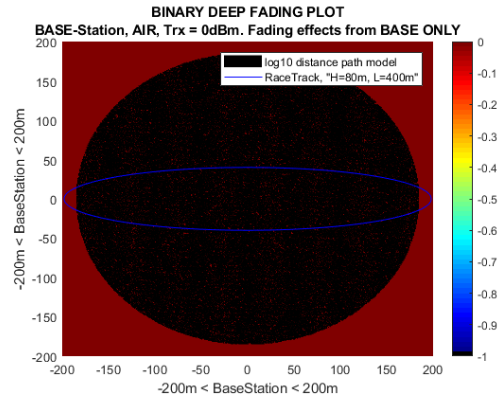

Figure 14: *Base station deep fading plot after base station has experienced fading.*

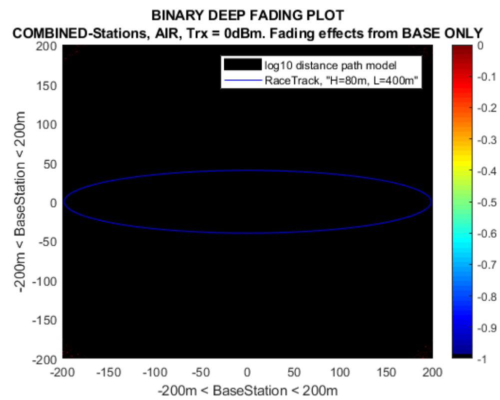

Figure 15: *Combined stations deep fading plot after base station experienced fading.*

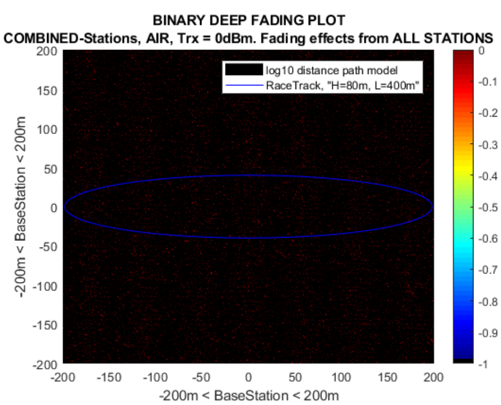

Figure 16: *Combined stations deep fading plot after all stations experienced fading.*

iii. Protocol decision example

Figure 17 shows the base station's RF input power of a run around the track from start to finish and figure 18 shows it for 47 rounds, which equals a marathon, and each round has different fading. Figure 19 shows the in-range of the base station packets which needs to be relayed or not.

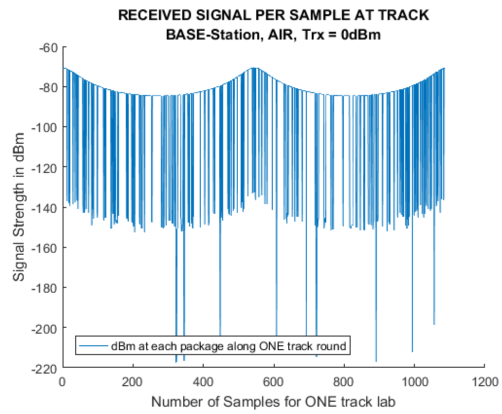

Figure 17: *dBm plot of a single run around the track.*

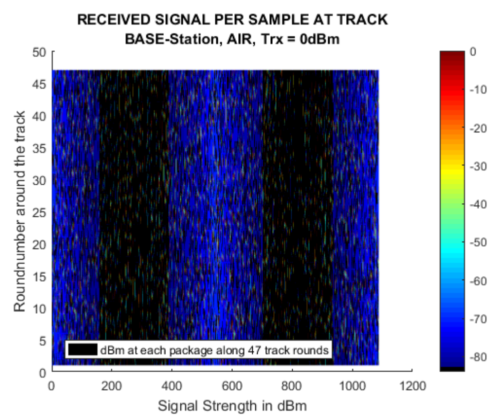

Figure 18: *dBm plot of 47 runs (marathon) around the track.*

A multi linear regression was fitted on the simulated data with three predictors: Distance, signal strength and whether the packet had been relayed before. The goal was to determine if the node should relay the next packet or not. Since signal strength and distance are strongly correlated in our simulation, due to antenna approximations and the binary behavior of the fading, they cancel each other out, while the

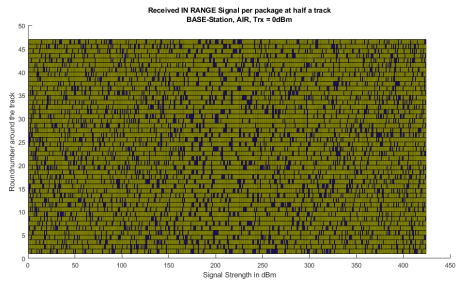

Figure 19: *In, half a track, base station RF input power transmitting range ToHopOrNot plot. Blue is relayed packages while yellow is direct package.*

previous packet status shows a 10% likelihood of the next packet needing to be relayed. Again, it is expected since the simulations have fading behavior added as a random variable appearing with a 10% likelihood. A real-life trial would be interesting, but is out of scope. Figure 20 shows the regression plot with the linear equation added in a legend box.

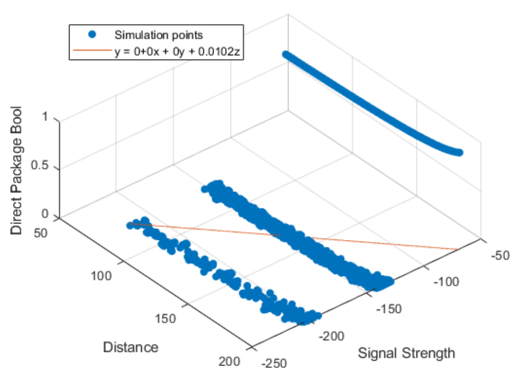

Figure 20: *Regression plot of the three variables: Distance, signal strength and if previous packet was relayed or not. The regression line is representing a change from direct transmission to relaying.*

iv. ARQ stop-and-wait method

The ARQ method is a data link-situated telecommunications scheme between two devices. It ensures that information is not lost due to signal fading, infused noise or other network failures and that frames arrive in a correct order. Multiple types of ARQ includes stop-and-wait, go-back-n and selective-repeat. In this project we will be using stop-and-wait because we only transfer a single packet (a heart rate measurement) from the runner and not a large file, i.e. an image. We will now look at the fundamentals of the method. We assume a network that consists of two nodes, n_1 and n_2, with n_1 sending some sort of data to n_2:

1. n_1 wants to send frames with data d_i to n_2. It prepares the first frame d_0 and transmits it to n_2. As soon as its done sending, n_1 starts a timer and expects to get an acknowledge frame (ACK) back from n_2 within that time telling n_1 that the frame has been correctly received.

2. n_2 receives the frame and sends a ACK frame back to n_1. Now n_1 can prepare the next frame $d_0 + 1$.

3. In case n_2 fail to acknowledge the frame in time, perhaps due to a network glitch or a faulty frame at the receiving end, the timer at n_1 will simply run out and it will retransmit frame d_0.

Aside from the ARQ stop-and-wait, we add a redundancy check number or parity bit (0 or 1) to all frames. The receiving node uses this number to verify the integrity of the frame, and does only send back an ACK if the frame passes this test. This adds a level of error-correctness to the ARQ method and helps avoid passing malformed data frames around. Two possible pitfalls exist with this version of ARQ: If the transmission medium has a long latency, the sender's timer could run out before the frame reached the receiver, and if the ACK sent by the receiver is damaged, the whole frame would have to be retransmitted. In both cases the receiver gets the same frame twice. One could use the parity bit to recognize duplicate frames, hence solve these problems. In terms of throughput, stop-and-wait falls by the wayside to go-back-N and selective-repeat because each frame has to be acknowledged separately, but may prove more useful in a noisy environment.

v. Energy calculations

In every wireless network system energy consumption is a must to evaluate. Based on measurements, see section v, calculations were made. Only lifetime of the base station is done in theory, and not individual hardware components as part of the node. It is assumed, that every packet is send directly to the base station successfully and a response is returned immediately. Total latency between the two nodes is set as constant at $12ms$ and an overshoot time at power-up from sleep mode is also constant at $50ms$ and 40% extra energy usage related to receiving energy.

The overshoot after wakeup is modeled as a Gaussian function exhibited in figure 21 for max energy wakeup. Table 1 shows the lifetime of the base station at six different scenarios all assuming the node has two full AA-batteries[6]. There are many ways of defining a network lifetime. In the scenario studied in this report, standard ways of defining wireless sensor network lifetime, like time until first node failure or first package loss, from the book[7], cannot be used. The lifetime expectancy is defined as half the battery capacity of a node. Calculations from this section can be found in appendix 9:

Scenario 1: Full power at transmission and otherwise always listening for packets.
Scenario 2: Min. power at transmission and otherwise always listening for packets.
Scenario 3: Full power at transmission, only listening for packets when receiving and no overshoot.
Scenario 4: Min. power at transmission, only listening for packets when receiving and no overshoot.
Scenario 5: Full power at transmission, only listening for packets when receiving and overshoot at power up.
Scenario 6: Min. power at transmission, only listening for packets when receiving and overshoot at power up.

Even though it costs to power up the node from sleep mode, it would still extend the battery life span putting it to sleep as often as possible.

[6] [WikipediaBatteryAA()]
[7] [Karl and Willig(2006), p. 65]

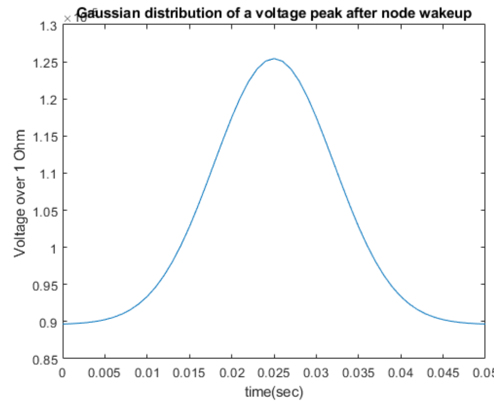

Figure 21: *Start up voltage peak after sleep mode.*

#	Power [dBm]	Sleep	Overshoot	Lifetime [Hours]
1	0.0	No	No	58.83
2	−25.0	No	No	58.54
3	0.0	Yes	No	4367.00
4	−25.0	Yes	No	44396.00
5	0.0	Yes	Yes	265.44
6	−25.0	Yes	Yes	265.70

Table 1: *Half capacity battery lifetime table for the base station at different transmission powers and with-/without sleep and overshoot.*

III. Implementation

This section describes how the theory discussed in section II has been implemented in our solution. We have divided the chapter into a overall section about the ARQ protocol and then sections per node, that is base station, the relay nodes and the runner node. For better readability, code snippets of the actual implementation has been turned into pseudo code. To view the nesC code please see the appendix 4, 5 and 6.

The chapter end with the section Energy Lab. In this section energy consumption will be discussed at different send/receive scenarios.

i. Generic ARQ implementation

The ARQ stop-and-wait method has been implemented on all nodes in the system. Figure 22 is a sequence diagram of the protocol implementation and shows how data requests are initiated by the base station and forwarded either directly to the runner (left side) or via a relay station (right side). Adding to this is the parity bit calculations to verify the integrity of received frames. This is conduced when a frame is received in either base station or one of the other nodes. The basics of the communications between the nodes are as follows:

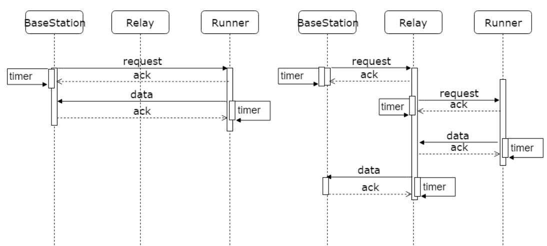

Figure 22: *The flow of data and acknowledgment messages between the nodes.*

1. The base station requests data at a certain time interval defined by a timer. When the timer runs out, it decides whether is should send the request directly or via one of the relays by using our protocol. To the request is attached a counter ($n = 0, 1, 2, 3$ -> n) and a sequence number (0 or 1) used by the receiver to verify the frame. When the radio is ready to transmit, it is forwarded to a specific node and the ARQ timer starts. It it now the responsibility of the receiving node to carry on the request. The receiver will send back a ACK frame to the base station if validation succeeded.

2. If the request reached the runner first, it will use the attached counter to calculate a new parity bit and compare this to the one in the request. If they match, it sends back an ACK followed by a data message containing the current heart rate of the athlete. A counter and a sequence number is attached to this as well.

3. The base station receives the data message, checks the parity bit, and saves it. It has now successfully obtained the heart rate.

4. If the base station decide to relay the request, the same procedure applies for the relay nodes. It is now the job of the relay to acknowledge the request, communicate with the runner using ARQ, obtain the data and send it back to the base station.

With ARQ and parity bit checking we archive a simple but reliable data-link connection between nodes and a way to discard damaged frames. One particular area of concern was the configuration of the timers, as they have to be consistent throughout the network. Obviously when relaying, the base station must take into account the turnaround time and possible retransmissions of the relay and the runner node before sending a new one. The base station timer (Ba_t) must be larger than the relay's (Re_t) and the runner's (Ru_t) combined, so $Ba_t > Re_t + Ru_t$ for the protocol to work correctly.

ii. Base station

The base station is the master node in our setup. It is responsible for three import tasks: Initiating data requests, deciding if the runner is out of range and keeping track of past events. Figure 22 shows the overall flow of data, but we shall examine the more detailed parts here. The main control loop is started by a timer every s second. It asserts if the runner is deemed out of range by calling a function and uses the feedback (either true or false) to increase an error counter that is used to change destination node. In other words, when a certain number of errors have occurred on the current link, we change request destination (from directly to relaying or vice versa). Listing 1 is an example.

```
Timer0.fired() {
  if out of range and link is direct {
    if the error count is below max
      use_next_destination
    else
      increase_error_count
  }
  if link is direct
    send_a_message_direct
  if link is relay to node north
    send_a_message_to_north
  if link is relay to node south
    send_a-message_to_south
}
```

Listing 1: *Main control loop of base station.*

Next is how the base station determines if the runner is out range. When new replies are received directly from the runner (it starts in range) we save the received signal strength indication (RSSI[8]) value in a first in first out queue with a length of 10. In such a queue, new data is inserted at back and taken out from the front. This means that the latest RSSI value of the runner is the back entry, with $n = 0, 1, 2...8$ being previous positions. Our algorithm takes a mean of these and multiplies it with a weighted score. If the latest position is lower than the mean, it is added to the weighted mean of the previous positions. If it is greater the average is then subtracted from it. The result constitutes a new estimated position of the runner. Listing 2 is an example.

```
bool isOutOfRange {
  if current size of queue is not max
    return
  for(i = 0; i < queue_size; i++)
    previousPos += queue_part[i]

  lastPos = queue_back_entry
  mean = previousPos/queue_size

  if lastPos is less than mean
    newPos = (lastPos*1)+(mean*0.1);
  else
    newPos = (lastPos*1)-(mean*0.1);

  if(newPos is larger than threshold)
    return true;
  return false;
}
```

Listing 2: *Out of range function in base station.*

The weights can be changed to put more empathize on the previous positions or more on the latest. Resulting on it acting more or less fast on new network conditions based on newest received packet.

iii. Relay

The relay stations in this setup are slaves to the base station. Their functionality is to relay messages to the runner node, where the base station node is out of reach. Because of this the relay station listens to the network and in case of being spoken to, it replies with an ACK and then sends a request for data to the runner node. In case it is not able to receive contact from the runner after three times, it sends an error message to the base station.

```
Receive.receive(message_t msg, uint8_t
    len){
if len == sizeof(requestMessage) {
  requestMessage reqmsg = (
      requestMessage) payload;

  if reqmsg_relayNodeid is TOS_NODE_ID
      {
    // Send Acknowledge
    sendAcknowledge(reqmsg);

    if reqmsg_data = 0 {
      requestFromBase = reqmsg;
      call Timer0.startPeriodic(
          TIMER0_PERIOD_MILLI);
    }
    else {
      requestFromRunner = reqmsg;
      call Timer1.startPeriodic(
          TIMER1_PERIOD_MILLI);
    }
  }
}
```

Listing 3: *Receive message event of Relay.*

In listing 3 the primary functionality of the two relay stations can be seen. The relay stations are always in need of commands to them before sending a command themselves, they will not work on their own and therefore relies on requests from the base station and answers from the runner.

iv. Runner

The runner node's task in our WSN is to respond to data request messages sent from the base station or the relay nodes, as seen in the sequence digram in figure 22. The node contains various settings that can configured as constants, so they are easy to change eg. the transmit channel and the radiation power of the antenna. When it receives a packet it will check that the packet was intended for the runner and if so it will send an ACK to the requester and get ready to send the data as a subsequent reply.

[8] RSSI is a scalar register value on the CC2420 radio calculated from the RF input power in dBm.

```
Receive.receive(Message pkt){
  if pkt is requestMessage {
    if request is for runner {
      send_Acknowledgement
      Timer_sendDataTorequester
    }
  }
}
```

Listing 4: *Runner receives requests and responds.*

The data contains the heart rate of the runner and in this scenario we just return a constant value. When sending data to the requesting node, it will start a timer and if it does not receive an ACK within a fixed time, it will resend the message three times followed by giving up on that reply.

```
sendData() {
  if If is not AntenaBusy {
    responsePacket_pulseData add
        runner_heart_beat
    if AMSend_send(responsepkt) is
        SUCCESS {
      AntenaBusy to TRUE
      resendCounter++
    }
  }
  if resendCounter is bigger then and
      equal to TRIES_TO_RESEND {
    resendCounter = 0
  }
}
```

Listing 5: *Runner sends data packet with runner's heart rate.*

v. Energy lab

To test the energy consumption of the nodes, a laboratory test has been conducted on two telosB units. On figure 23 the test setup is defined. The test was conducted with the same method as laboratory exercise 5[9]. Data and pictures can be found in appendix 7, codes for conducting the laboratory experiments can be found in appendix 8.

The measurements can be seen on figure 24, 25, 26 and 27. Inspecting the values of the three signal strengths on figure 24, 25, 26 and 27, and converting these voltages to a current, provides the results found in table 2.

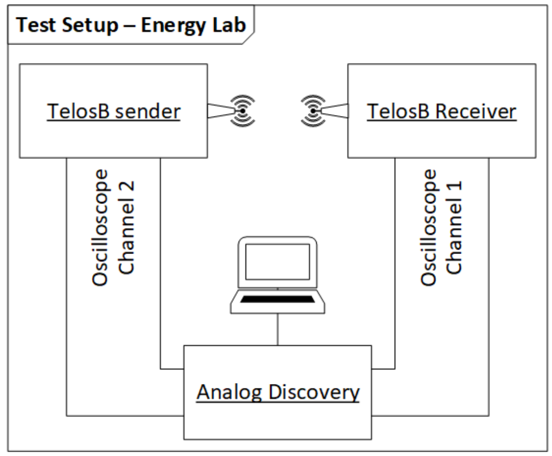

Figure 23: *Energy lab test setup.*

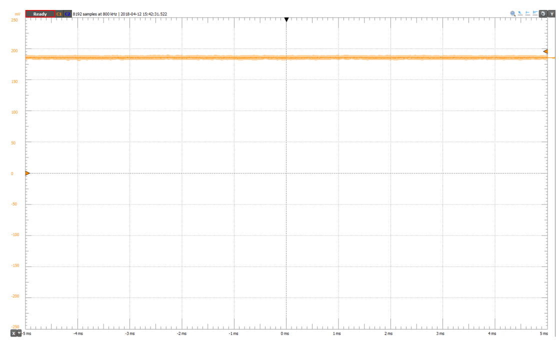

Figure 24: *Voltage drop at resistor $R = 10\Omega$ in series with Receiver (yellow), radio on, doing nothing.*

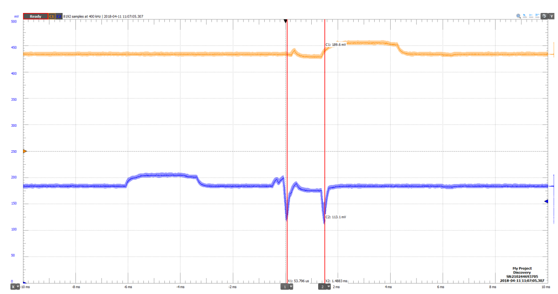

Figure 25: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at* $0.0dBm$.

[9] [Madsen(2018)]

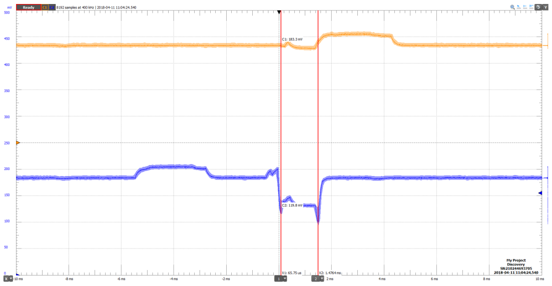

Figure 26: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at $-12.5dBm$.*

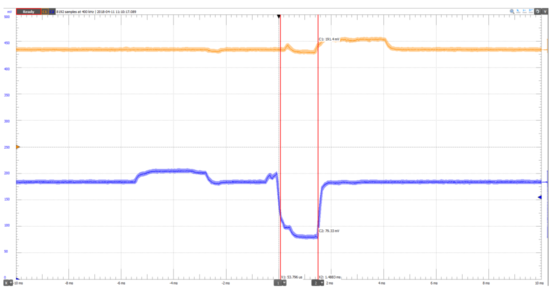

Figure 27: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at $-25.0dBm$.*

Radio	Voltage [mV] at $R = 10\Omega$	Converted Current [mA]
OFF	3.32	0.33
Listening	182.24	18.22
Sending $0.0dBm$	172.31	17.23
Sending $-12.5dBm$	123.30	12.33
Sending $-25.0dBm$	87.20	8.72

Table 2: *Mean current drawn from the telosb calculated with voltage drop at resistor R.*

Seen in table 2, there is a big difference in how much current is drawn at different signal strengths. However at further inspection of the telosb units, it is concluded that turning the radio on, will make a unit automatically start listening for wireless signals. This results in the units always drawing a lot of current whenever the radio is turned on.

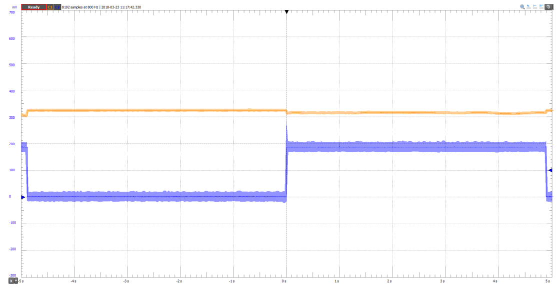

Figure 28: *Voltage drop at resistor $R = 10\Omega$ in series with telosb unit (blue) and battery (yellow), radio switching between on and off.*

As seen in figure 28, turning on the radio is indeed expensive compared to having it turned off. At first thought one might design a protocol that turns off the radio when not needed and on when needed. This will require a protocol with time/clock synchronization of some sort to be implemented, and will not be implemented in this report. An overshoot can be observed at radio turn on, this means that it will cost an extra amount of energy whenever the radio is turned on. This concept have already been talked about in section v in which the data came from this section.

IV. Test and performance

This section describes the test configuration and how we measured the performance of our protocol.

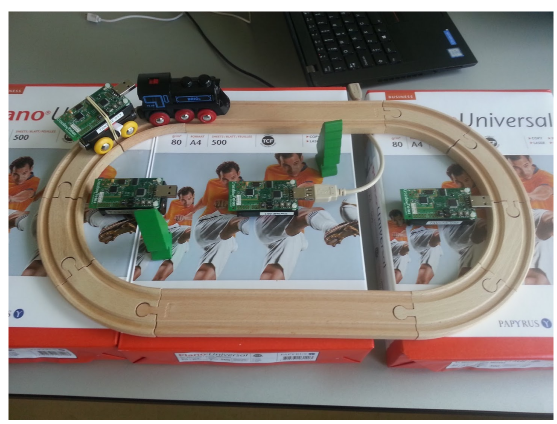

Figure 29: *A BRIO toy train track with the base, north and south relay stations in the center and the runner put on top of the train.*

i. Setup

To test our original scenario described in the introduction section I on page 1, we built a smaller version of the running track as a toy train track. To get a constant speed of the runner we used a BRIO train to symbolize a human runner that runs a track. The speed of this individual was set to $12\frac{km}{hr}$, but the BRIO train had a speed of $0.2604\frac{km}{hr}$.

The track we used in the theory chapter II was $906.17m$ long and the runner would run a marathon which is $46.56km$ and therefore equals 46.56 rounds. Using the standard BRIO track pack we build a track with length of $1.12m$. In the theory we send $4\frac{packets}{sec}$ and that means our time per packet should be $\frac{1}{4sec} = 0.250ms$, thus about 1087 packets per round if the runner runs at an average speed of $12km/h$. But as our train is vastly slower than an actual runner, and the testing train track was shorter than the running track, we needed to account for the differences between our test setup and the theory we had calculated. Our initial calculations showed that we needed to send packets with a frequency of $14ms$ and that was not possible due to limits of the hardware. As seen in figure 27 it takes $10ms$ to send a packet without accounting for other tasks going on in the telosb node, e.g. handling computations. We divided the packets per round by 4; $\frac{1087}{4} = 272$. As a result needing to send a packet every $(\frac{272}{15.5})^{-1} = 57.016ms$. It takes the train 15.5 seconds to run one lap. A picture of the setup can be seen in figure 29. For detailed calculations, please see appendix 3.

V. WiFi conditions

In our test we used two different WiFi channels (4 and 11) and we found the best and the worst by using the WiFi Analyzer app[10] from the Android app store. As seen in figure 30, channel 11 is used by Aarhus University WiFi, but channel 4 looks rather free, so we picked both as seen in table 3. The -40 RSSI value is calculated from the telosb data sheet[11] by taking the stated receive sensitivity of the TPR2420CA chip ($-94dbm$), subtracting $-10dbm$ to accommodate for noise and converting to CC2420 RSSI by subtracting a offset of -45[12].

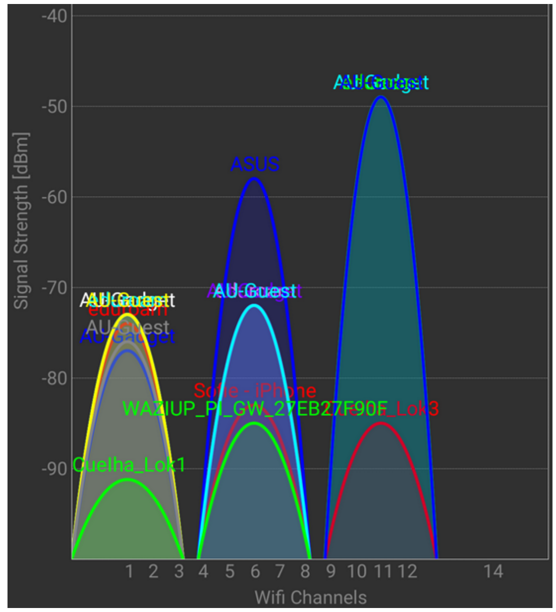

Figure 30: *Signal strength of WiFi channels on the test day. Channel 11 was busy and channel 4 was a quiet channel.*

We ran 4 tests for 48 minutes each testing our protocol with different channels and track lengths.

Scenario	Channel	RSSI	Length of track
1	11	-40	$43.5cm$
2	11	-40	$56cm$
3	4	-40	$43.5cm$
4	4	-40	$56cm$

Table 3: *Test scenarios performed.*

VI. Results

Using the test setup and mechanics described in section IV, we conducted a experiment of four different scenarios. We wanted to see whether the hopping frequency would increase, when we increased the circumference of the track, evaluate the packet sent/loss ratio and measure the overall perception of data. The test results would indicate if our relaying algorithm, the implemented stop-and-wait ARQ

[10][Farproc@gmail.com(2018)]
[11][Crossbow Technology Inc.(2004)]
[12][Chipcon Products()]

protocol and distance measurements were correct and working. All results are gathered from the base station and can be found in appendix 10. To quickly iterate the parameters measured in the test runs:

***n* request packets sent (p_1):** Number of packets sent from the base station over the course of the test. These include retries due to ARQ timers expiring. Read this number as "times we have asked for data".
***n* packets relayed to node 1 (p_2):** Number of packets relayed to north relay station.
***n* packets relayed to node 2 (p_3):** Number of packets relayed to south relay station.
***n* ACK's received (p_4):** Number of acknowledgments received by the ARQ protocol. The closer this number is to the packets sent, the more stable the data link connection between our endpoints is.
***n* DATA's received (p_5):** Number of data packets received. Ideally, this should be close to the packets sent as well. If $p_5<p_1$, then p_1-p_5 request packets were not replied.
***n* packets not acknowledged in time (p_6):** Number of packets not acknowledged before the ARQ timer ran out. This is strongly related to the timers on the base station and heavily influenced by interference and signal noise. Also in our test set-up we tried to get as much data from the runner as possible, so timers were strict.

Table 4 and 5 show the final result of each completed scenario lasting 48 minutes each, which equals about 168 (small track) and 130 (large track) rounds with the train. Both tracks have the same amount of total packages sent, 12280. Each minute we recorded parameters p_1 to p_6. All are initially set to zero. We decided not to change the RSSI threshold in each scenario.

Sn.	p_1	p_2	p_3	p_2+p_3
1	29148	4066	3512	7578
2	29838	8115	8499	16614
3	29258	3706	4021	7727
4	30523	1232	8425	9657

Table 4: *Data from scenarios 1-4 for parameters 1-3.*

As expected, the results vary depending on the track length and the channel used. When using a length of

Sn.	p_4	p_5	p_6
1	22169	34167	20700
2	17880	23563	21111
3	21944	32830	21244
4	19137	28874	23188

Table 5: *Data from scenarios 1-4 for parameters 4-6.*

56*cm* (scenario 2 and 4), we see $p_5 \leq p_1$, meaning the base station received the same or less data packets than requested. In scenario four they are even fairly close. If $p_5 > p_1$, it could be due to fading or missed timers. Also worth noting is the increased use of relays (p_2 and p_3) when the length is 56*cm*. Changing channels between 11 and 4 seem to lower the difference between p_1 and p_5 as well.
Figure 31 shows the correlation between the number of ACK's received, p_4, at the base station per packet sent from it. These numbers should be close to each other. Scenario 2 received a large number of ACK's around the 24408 packet sent mark, perhaps due to deep fading.

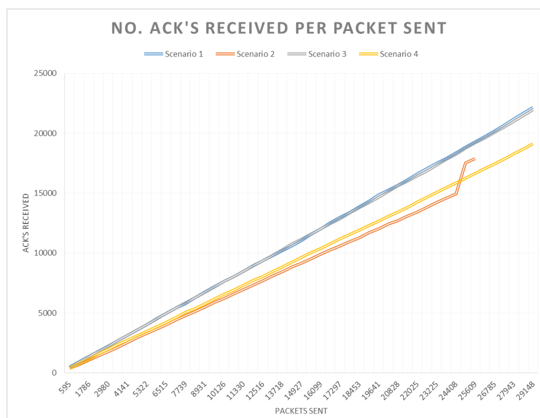

Figure 31: *No. ACK's received per packet sent.*

Figure 32 shows the number of data packets received, p_5, from the runner node either direct or via one of the relays per packet sent. Scenario 1 and 3 ($length = 43.5cm$) follows each other slightly, however changing channel from 11 to 4 decrease the amount of additional packets received ($p_5 > p_1$) when using $length = 43.5cm$. Scenario 4 looks to be the most successful one, as the number of data packets received is close to the number of requests sent. We reflect on this at the end of results. Scenario 1 and 3 sees more data packets being received than requested. A probable cause is a packet not being acknowledged in time by the base station, so the runner will resend it, even though it might already have been received by the base station.

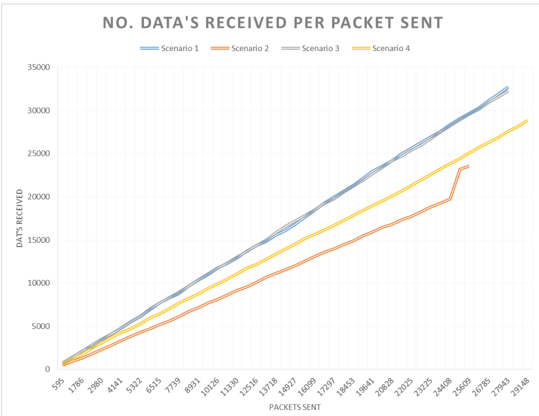

Figure 32: *No. DATA's received per packet sent.*

Figure 33 shows the number of data relayed to the north and south relay station combined per packet sent. As expected, scenario 2 and 4 ($length = 56cm$) relays more packets, now that the runner node will be out of reach from the base station for a longer period of time. It might even be that neither can reach the runner due to deep fading. Calculations made in Appendix 1 show the minimum theoretical packets send from the base station, which must be relayed at each of the track length. The total packets sent from the base station ideally is 12280, the minimum relayed packets send from the the base station for scenario 1 and 3 are 2355 packets, and for scenario 2 and 4 are 4961 packets. Both the number of relayed packets and the total number of packets send from the base station is flawed compared to theory and this will be further discussed in the discussion, but the inter scenario relative test results follow the theory well. The base station should send a request to the runner after a specified time to investigate if the runner has returned into direct communication reach, of the base station. Alternative the RSSI of the relaying node could be evaluated by the base station to give an idea of the runner node position.

Figure 34 shows the number of packets not acknowledged in time per packet sent. All scenarios follows each-other, which points to a systematically error

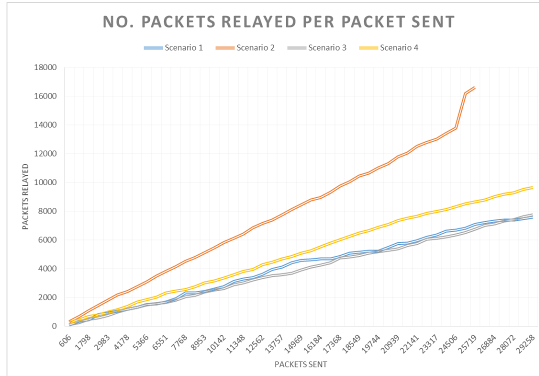

Figure 33: *No. packets relayed per packet sent.*

either due to strict timers or fading issues. An error that channel or length does not seem to affect. We noticed during the live testing that more ARQ errors occurred when communicating directly rather than relaying. Figure 35 and 36 show how the two scenarios that relay the most packets (2 and 4) distribute the data among our two relay stations (north and south) per packet relayed. A bit strange is the sudden drop of packets relayed to the north station in scenario 4, figure 36. Probable reason is fading.

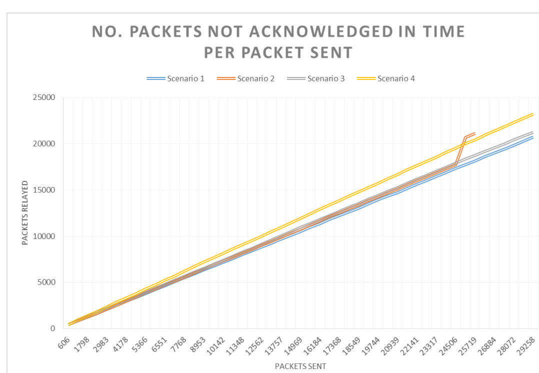

Figure 34: *No. packets not acknowledged in time.*

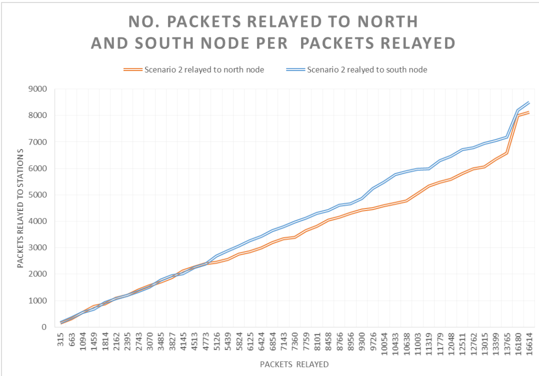

Figure 35: *No. packets relayed to north and south node in scenario 2.*

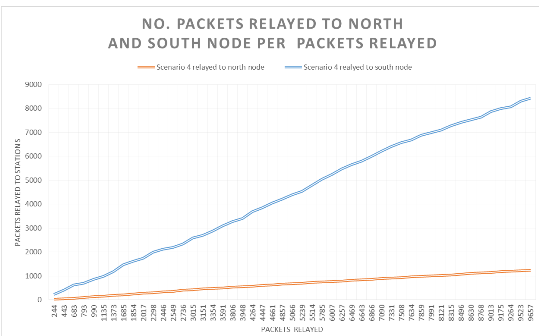

Figure 36: *No. packets relayed to north and south node in scenario 4.*

In terms of event quality (number of packets requested (p_1) vs. received (p_5)) the different scenarios has various characteristics. A optimal scenario is one where the number of requests sent is close to the number of data packets received, that is $p_1 \approx p_5$. If $p_1 > p_5$, we have asked for more events than received, i.e. packets have not been send or lost. If $p_1 < p_5$, we have received more events than asked for, i.e. information overload. Table 6 shows event quality for each scenario (1-4). Surprisingly scenario 4 provides the overall best event quality, and not scenario 3, i.e. using a channel with less traffic and increasing the length of the track, hence relying more on the relay stations.

Looking at the energy consumption, the smaller the track the less total energy will be consumed because of less messages relayed. This result in less total energy consumed because the relays are inactive, assuming the relays are sleeping when inactive. However this is not true in the test scenarios, but in a optimized scenario this would be the case and is therefore concluded upon. Energy consumption per packet sent can be found in table 7. As shown, the total energy consumption of the system as well as that of the base station itself is higher in scenario 2 and 4 because of the extra request messages send through relays. The conclusion is that the more messages or packets that needs relaying the more energy is consumed overall. Surprisingly again scenario 3 was not the optimal setup, but it turned out to be scenario 1.

Sn.	Requests sent	DATA's received	% Quality difference.
1	29148	34167	17.2%
2	29838	23563	−21.0%
3	29258	32830	12.2%
4	30523	28874	−5.4%

Table 6: *Event quality from scenarios 1-4.*

Sn.	Total energy [Ah]	Base station percentage used [%]
1	0.577	8.81
2	0.731	9.02
3	0.581	8.84
4	0.632	9.23

Table 7: *Energy consumption assuming a specific cost per packet sent. "Total energy" is the energy consumed by the whole system. "Base station percentage used" is the percentage of the total energy used by the base station.*

The result of both event quality and energy consumption show that the size of the track matters. Increasing the size of the track and adding more relays improves the event quality but also increases the energy consumption. It all come down to a tradeoff between event quality and energy consumption. Track size results show that the effect on event quality overshadow the drawback of intensified energy consumption when increasing the track size.

VII. Discussion

In the test scenario we see a lot more packets sent than what we in theory expected, as calculated in section VI Results. It is hard to confirm the exact problem, however because the nodes are sending at speeds of 20ms, it is easily assumed that there are problems with the hardware not having time to send acknowledges before a new request is send. This is the results from trying to emulate the report scenario in a smaller scenario with the train track, where the train speed does not comply to the other factors of the field, resulting in too high a requirement of the node hardware, to be able to give theoretical results. Changing to a correct scenario of a runner on a running track, will give very different results.

Under the test, it is shown that there is a higher percentage number of relayed messages than what was assumed in theory, as calculated in section VI Results. The reason behind, is that the protocol was implemented with the knowledge of understanding when to change from direct to relay messaging effectively, but not the other way around. In short the base station knows when to change from directly talking to the runner node to relaying the message through a relay node, but does not get information on when to change back again before the relay is out of reach of the runner node.

There are a few things that could have been done to improve various parameters in this project. Some could be entire projects in their own. To improve battery life of the relays stations and the runner, one could implement a medium access (MAC) protocol such as the S-MAC that make nodes periodically sleep and auto-synchronize their sleep schedules. As we have shown, solely using the antenna uses a lot of energy and the S-MAC would help mitigate challenge in a WSN. Because of the way our case works, we have an estimated position of the runner at time t, which we can exploit to our advantage. When the runner is at the north area of the track therefore being covered by the north relay station, it makes little sense to have the south relay turned on and consuming energy. A way to optimize this would be to send a message to the relay not in use and make it turn off its antenna and enter sleep mode, then wake it up later when the runner is within range of that station.

If we made the relay stations able to relay between each-other then that would increase the scalability of the network. The stations would be able to cover a even bigger track if they were relaying messages by a greedy algorithm pattern always using the shortest path to the runner.

Our protocol in the current state is susceptible to attackers who would take advantage of the fact that our base station will keep using a relay if it can continue to provide heart rate data. The attacker could abuse this by performing a man-in-the-middle attack acting as a legitimate relay station and either just eavesdropping, drop the packets or falsify contained information.

When looking at the energy consumption, we can conclude that the best method of saving energy is to shut down the radio when not needed, if it can be off in enough time that the overshoot, when the radio is turned on, will not be more expensive than having the radio on at all time. There are also ways of acquiring energy rather than just saving it. In this scenario using solar energy and passive human power generation methods are viable, since the nodes are outside at day and the runner node is attached to a human.[13]

VIII. Conclusion

The focus of this project has been to build a WSN protocol that would relay packets based on accumulated transmissions errors and the received signal strength from RF input power measurements. The protocol was designed for a real-life running track with a base station actively requesting heart rate data from a runner carrying a sensor node, however for testing purposes we scaled parameters including track length, distance between nodes and the request interval to match a practical arrangement of a toy train driving a short oval-shaped track.

We concluded on the different scenarios, air versus engineering building and signal strength of $0dBm$ versus $-24dBm$, that the test results align according to derived theory.

Using two different WiFi channels, a clear and a noisy channel, we conclude that the implemented protocol, relative to signal strength and transmission error, had an positive effect on number of relays utilized. It was especially observed that the results on the increased track size were improved on a clear WiFi channel compared to the noisy channel.

[13] [Sudevalayam and Kulkarni(2011)]

Our results show, that the protocol is able to relay packets when required to reach the runner node, but not all packets gets acknowledged in time by the data-link ARQ protocol built on top. This is possibly due to strict timings on the sender's part in the test scenario.
We show with conducted measurements that the antenna (CC2420) can be very energy-consuming, while processor computations on the telosb are far cheaper. Therefore we conclude, that packet hopping should be kept at a minimum whenever possible to avoid additional energy dissipation.
Regarding the use of RSSI to relay packets, we echo the thoughts of [Heurtefeux and Valois(2012)] and find that the readings from CC2420 can be fairly inaccurate and do not represent a clear picture when predicting whether the next packet in a data stream will be lost, thus additional features should be included in the decision-making.
The final conclusion, to hop or not to hop, it all comes down to a trade-off between event quality and energy consumption, like all other wireless sensor networks. The protocol and amount of wireless sensors need to be defined, based on the task at hand. An energy optimized setup would be to use few relayed messages while event based optimized setup would be to increase the amount of relays in the system for better overall coverage.

References

[Chipcon Products()] TI Chipcon Products. *CC2420 2.4 GHz IEEE 802.15.4 data sheet*. Texas Instrument.

[Crossbow Technology Inc.(2004)] Crossbow Technology Inc. TelosB Mote Platform. pages 1–2, 2004. doi: 6020-0094-01Rev.B.

[Farproc@gmail.com(2018)] Farproc@gmail.com. Wifi Analyzer, 2018. URL `https://play.google.com/store/apps/details?id=com.farproc.wifi.analyzer{&}hl=en{_}UShttp://wifianalyzer.mobi`.

[Heurtefeux and Valois(2012)] Karel Heurtefeux and Fabrice Valois. Is RSSI a good choice for localization in wireless sensor network? *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, (December 2014):732–739, 2012. ISSN 1550445X. doi: 10.1109/AINA.2012.19.

[Ka'bi(2016)] Amin H. Al Ka'bi. On Frequency Tuning of Inverted F Antenna. pages 11–14, 2016.

[Karl and Willig(2006)] Holger Karl and Andreas Willig. *Protocols and Architectures for Wireless Sensor Networks*. 2006. ISBN 0470095105. doi: 10.1002/0470095121.

[Madsen(2018)] Jens Kargaard Madsen. Lab exercise - Energy & Power consumption. *Department of Engineering, AU*, 2018.

[Sudevalayam and Kulkarni(2011)] Sujesha Sudevalayam and Purushottam Kulkarni. Energy harvesting sensor nodes: Survey and implications. *IEEE Communications Surveys and Tutorials*, 13(3):443–461, 2011. ISSN 1553877X. doi: 10.1109/SURV.2011.060710.00094.

[WikipediaBatteryAA()] WikipediaBatteryAA. Battery, AA, RAM, Capacity under 50 mA constant drain 20180528at1231. URL `https://en.wikipedia.org/wiki/AA{_}battery`.

Appendixes

Appendix1
A pdf-file with various mathcad calculations. Range calculations, Doppler spread and Samples per round.

Appendix2
A txt-file with various matlab code. PowerCircle, PowerCircleOffice, FadingPlots, ToHopOrNotToHop and Protocol_Decision.

Appendix3
A pdf-file with mathcad calculations for the running track and the train track scenario.

Appendix4
A zip-file with the nesC code for BaseStation.

Appendix5

A zip-file with the nesC code for Relay.

Appendix6
A zip-file with the nesC code for Runner.

Appendix7
A zip-file with the different results from various energy laboratory tests.

Appendix8
A zip-file with the different nesC codes for various energy laboratory tests.

Appendix9
A pdf-file with matlab code for lifetime calculations.

Appendix10
An excel-file with results from train track scenario.