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

May 24, 2018

Abstract

Whether to relay data packets or to send them directly between two computers in a wireless network setting is a discussed topic in computer engineering. Typically, when a stationary wireless router or data sink is out of reach of a consumer in a wireless local area network (WLAN), other access points in place must relay the packets of the consumer. In a wireless sensor network (WSN), addtional concerns come into play, such as energy usage, battery levels and noise interfence, which must be considered before relaying packets. In this project we establish a WSN of telosb nodes of one base station, two intermediate relay nodes and a destination node being carried around a oval-formed track by a marathon runner. The base station will continously communicate with the moving node either directly or via the relay nodes. We design a relaying protocol that will take into account signal strength when deciding if it should relay a packet or not. We conclude, that using signal strentgh values from the CC2420 radio can be unreliable when used in a real-time relaying protocol and that a wireless sensor node should only relay packets when it is absolutely necessary in order to save energy.

I. Introduction

This report details a project done in Wireless Sensor Networks (WSN) with a mini-project called "To hop or 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. This information is going to be transfered to a sink (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 signal strength and energy consumption into consideration. For inter-node connectivity, we will design and implement the data-link layer stop-and-wait ARQ method on all nodes.[1]

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.

[1] [Chipcon Products()]

i. Scenario introduction

A marathon runner is racing a track shaped as shown in figure 1, while equipped with sensor node A. It broadcasts four packets per second tracking the runner's heart rate. The level of details in the tracking package defines the number of marathons possible for the runner to run before a new battery is required. At one time the track was closed, resulting in the runner racing around the building of her workplace.

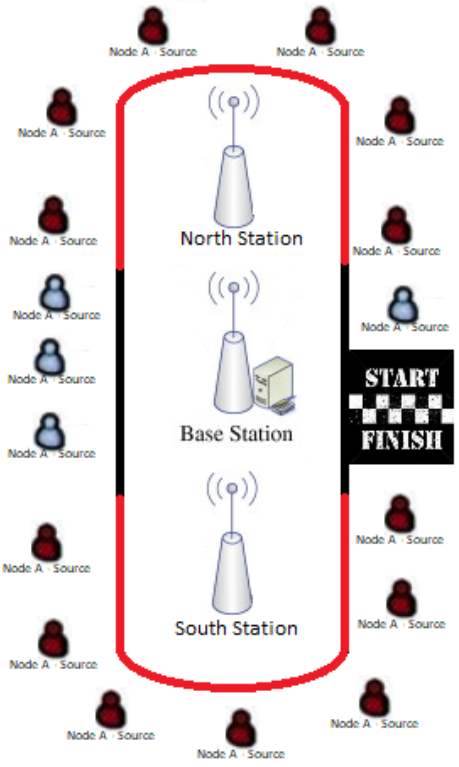

Figure 1: *A marathon runner "node A" is racing around a track, while transmitting heart rate information to the base station. In the red northern territory, node A transmits to the north station which relays the message to the base station. Likewise, at the southern station.*

ii. Protocol introduction: "To hop or not to hop"

The base station will be requesting data from the runner node with a packet size of 128 bytes and save it in storage. When the base station is out of range, the north or the south relay station will receive the request and relay it to the runner. Time Synchronization, Localization and Scalability will be considered regarding the protocol design. Each and combined scenarios will be evaluated in relation to signal strength relative to power consumption and data reliability.

II. Energy consumption

i. Runner node

The base station will be requesting data from the runner at a periodic transmission rate. Different levels of energy consumption will be determined by the chosen protocol, but in any case, the power consumption will be considered, over a period of one second, constant. The less energy consumption of the runner will give longer individual lifetime and runtime for it, but low signal strength of node A might not give a lowest possible system power consumption. Depending on the needed quality of the received package, e.g. -84dBm, a cut of distance will be calculated and measured. Distance measuring will be limited by interference providing a need for a scalable transfer function estimate and an average over multiple measurements. Life time of node A will be considered when half of the battery capacity is used.

ii. Base station, north and south relay station

Idle time, receiving and transmitting power consumption will be calculated and measure. When out of range the âĂIJpoleâĂİ stations will in theory go to an idle state to save power. When in transmitting mode different measurements will be conducted depending on the chosen protocol. E.g. firm or no handshakes between pole station and base station will be measured leading to different possible distances between jumps.

iii. Sink: Base Station

The required detail of information needed to give a good user estimate will raise the question of acceptable package loss. Signal strength, package frequency, package loss vs reliability from both pole stations and source will determine the power consumption of the base station and the system. The base station will never be in idle state and it must be able to reach pole station resulting in the highest cost function of the system.

III. 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 engineering 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 [6]. 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[2]. To simulate a real scenario, the transmitting antenna could be mounted on a rotational 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.

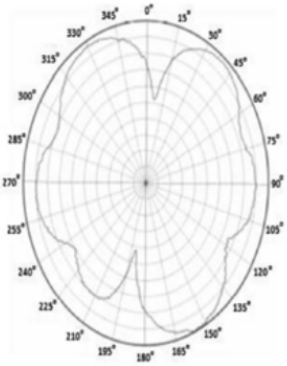

Figure 2: *Estimation of an inverted F-Antenna radiation pattern.*

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 cover. 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.

Lowering the transmission power can prolong the lifetime of individual nodes and the wireless network

[2] [Chipcon Products()]

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

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

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

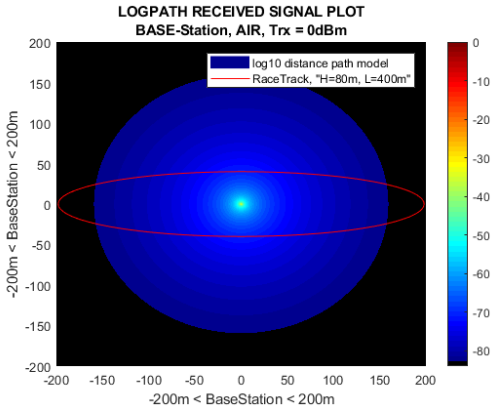

Figure 4: $-84dBm$ 159.3m *received signal area of the running track. The antenna cannot cover the entire track.*

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 page 93 [REF 1]. Figure 9 and 10 show the distance at minimum Ptr and inside Shannon providing a stunning 0.035% of the coverage related to full power in open AIR.

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 set the signal to noise ratio (SNR) floor of the received signal, but more expensive electronics can compensate for the former noise source, and if no mobile phones are at the track, it can lead to less of the latter noise source, 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 sink 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

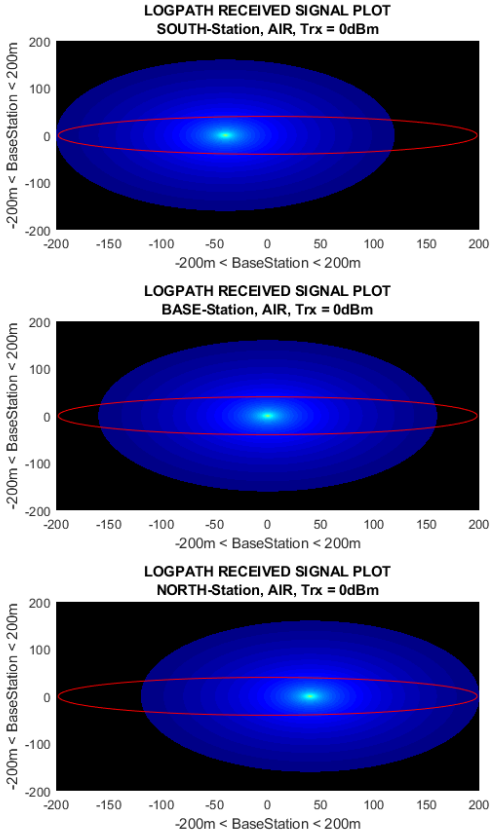

Figure 5: *Three stations covering the entire track individually.*

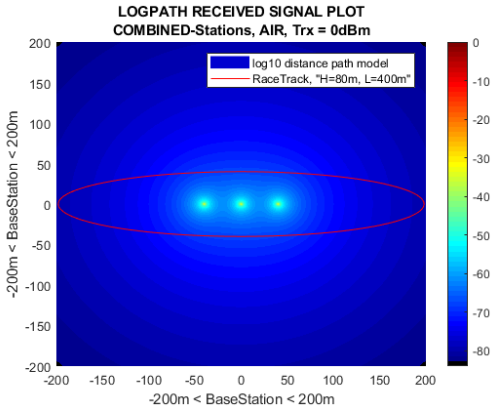

Figure 6: *Three stations covering the entire track combined.*

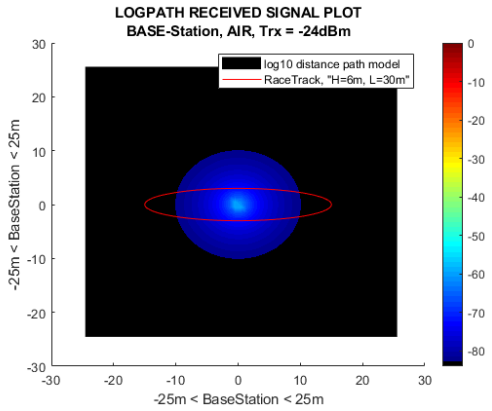

Figure 7: 10.1*m adequate RF input power for the base station.*

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 source and the sink 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 2 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

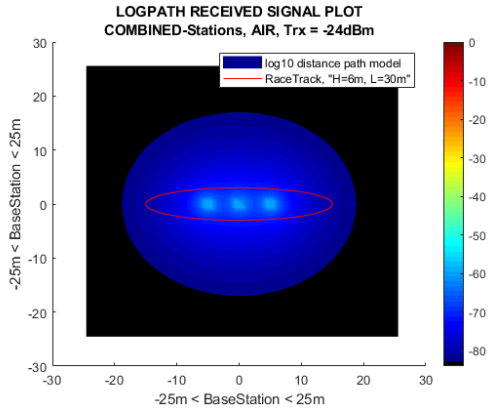

Figure 8: 10.1*m Three stations combined RF input power.*

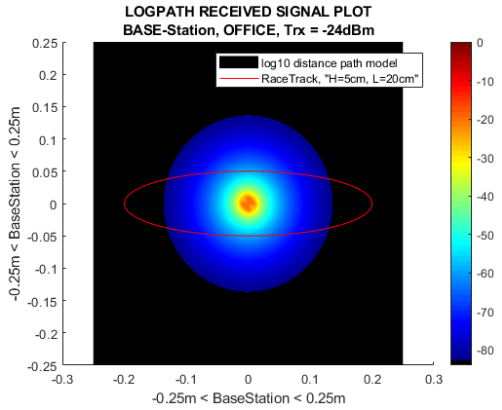

Figure 9: 13.1*cm adequate RF input power for the base station.*

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 sink and the NLOS signal travels $1.125m$ to the sink, 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

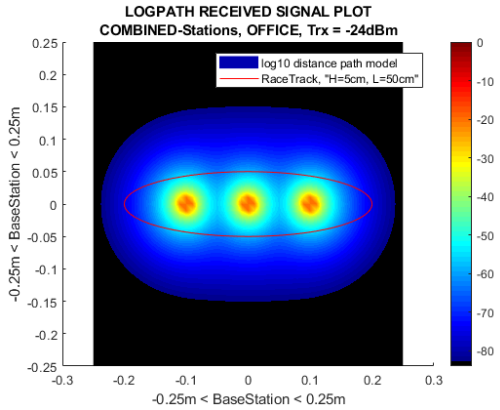

Figure 10: 13.1*cm Three stations combined RF input power.*

not have a full cancellation at the sink, since the LOS signal will have a higher amplitude than the NLOS signal. The drop-in signal strength can be estimated to around $30-40dB$ $(60-70dBm)$[3] 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 sink, e.g. more runners on the track or a photographer taking images 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.

iii. Protocol Decision Example

Figure 10 (left) shows the base station's RF input power of a run around the track from start to finish and figure 10 (right) shows it for 47 rounds, which

[3] Appendix: 1, page 92

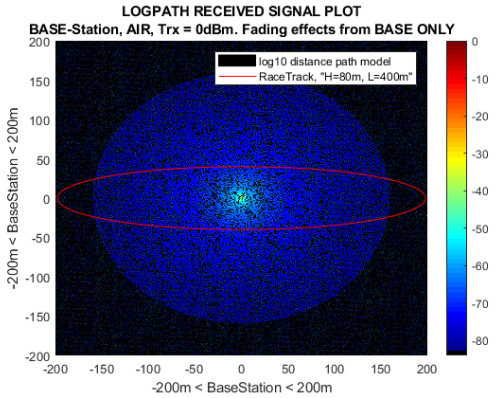

Figure 11: *Base Station RF input power plot after base station experiencing fading*

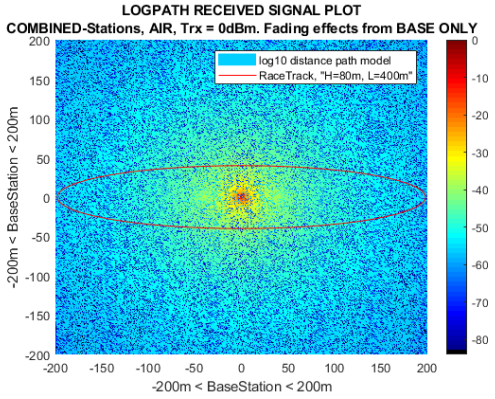

Figure 12: *Combined Stations RF input power plot after base station experiencing fading*

equals a marathon, and each round has different fadings. Figure 17 shows the in-range of the base station packets which needs to be relayed or not.

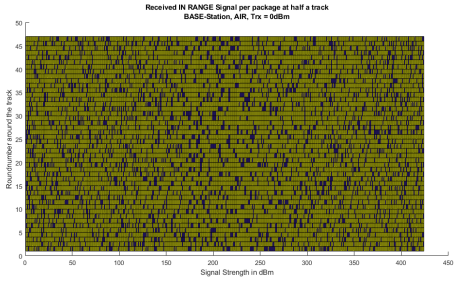

Figure 17: *In, half a track, base station RF input power transmitting range ToHopOrNot plot. Blue is relayed packages while yellow is direct package.*

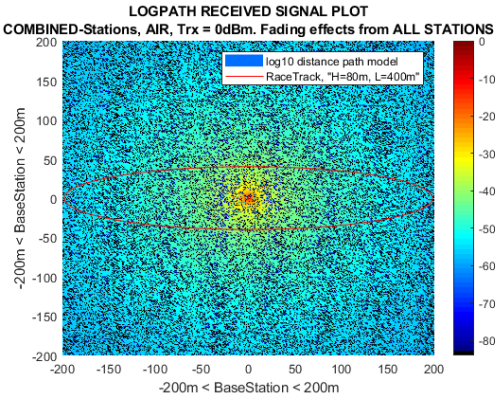

Figure 13: *Combined Stations RF input power plot after all stations experiencing fading*

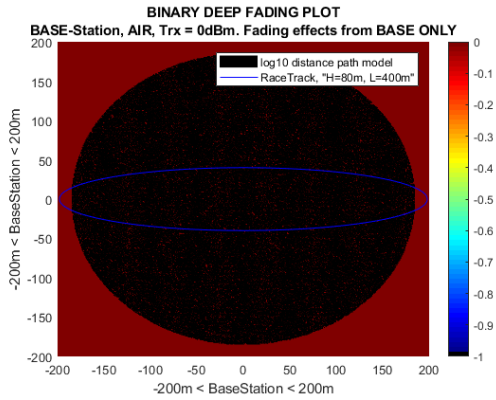

Figure 14: *Base Station Deep fading plot after base station experiencing fading*

A multi linear regression was fitted on the simulated data with these predictors: Distance, signal strength and whether the packet has been relayed before. The goal was to determine if the node should the next packet relay 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 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 18 shows the regression plot with the linear equation added in a legend box.

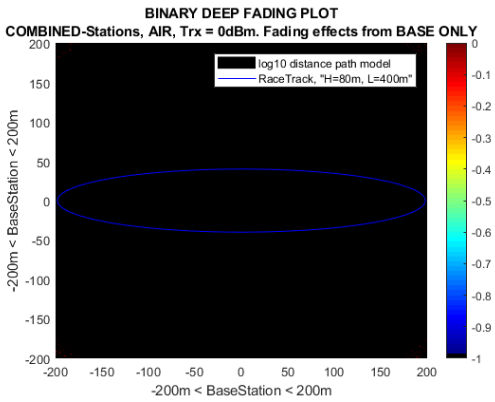

Figure 15: *Combined Stations Deep fading plot after base station experiencing fading*

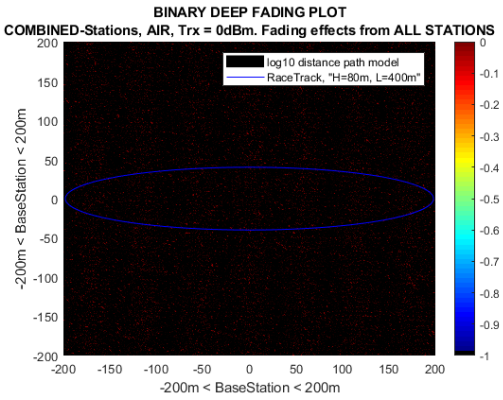

Figure 16: *Combined Stations Deep fading plot after all stations experiencing fading*

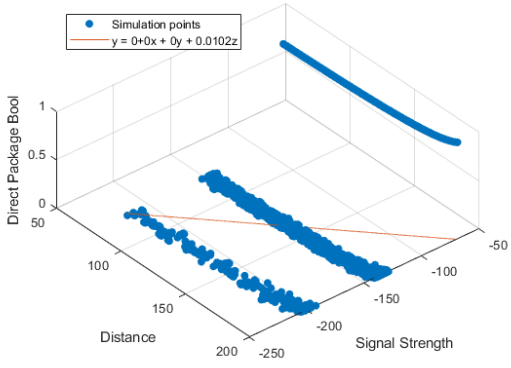

Figure 18: *Regression plot of the three variables: Distance, signal strength and if previous packet was relayed or not. The regression line is representing a change from direct transmission to relaying.*

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. Since energy consumption is not the main investigation of the project, only lifetime of the base station is done in theory. We assume, that every packet is send directly to the sink 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 19 for max energy wakeup. Table 1 shows the lifetime of the base station at six different scenarios all assuming two full AA-batteries:

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

Calculations are in appendix ??????. 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.

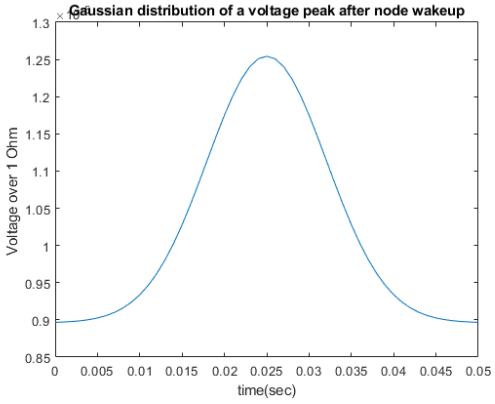

Figure 19: *Start up voltage peak after sleep mode.*

#	Power [dBm]	Sleep	Overshoot	Lifetime [Hours]
1	0.0	No	No	176.48
2	−25.0	No	No	176.52
3	0.0	Yes	No	13101.00
4	−25.0	Yes	No	13319.00
5	0.0	Yes	Yes	796.32
6	−25.0	Yes	Yes	797.11

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

IV. Implementation

This section describes how the theory discussed in section III 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.
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 20 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.

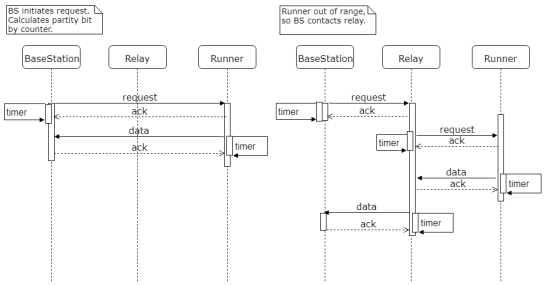

Figure 20: *The flow of data and acknowledgment messages between the nodes.*

The basics of the communications between the nodes are as follows:

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, decide if the runner is out of range and keep track of past events. Figure 20 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[4]) value in a FIFO(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 used 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 nodes 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 node listens to the network and in case of being spoken to, will reply with an acknowledge and further send data to the Runner node. In case that the Relay node is not able to get in contact with the Runner node after three times, it sends an error message to the Base station. The Relay nodes are always in need of commands to them before sending a command themselves, they won't work on their own and therefore relies on the Base station.

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

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 20. 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 acknowledge (ACK) to the requester and get ready to send the data as a subsequent reply.

```
Receive.receive(Message pkt){
if(pkt == requestMessage) {
//Check if the request is for me and if
    it is send sendAcknowledge.
if (pkt.nodeid == requestpkt.relayNodeid
     && requestpkt.data == 0) {
sendAcknowledge();
Timer.sendDataTorequester(); }
```

Listing 3: *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 (!AntenaBusy) {
    responsepkt.pulseData = currentpulse
        ();
    if (AMSend.send(responsepkt) ==
        SUCCESS) {
      AntenaBusy = TRUE;
      resendCounter++;
    }
  }
  if(resendCounter >= TRIES_TO_RESEND){
    resendCounter = 0;
  }
}
```

Listing 4: *Runner sends data packet with runner's heart rate.*

v. Energy Lab

To test the energy consumption of the motes, a laboratory test has been conducted on two telosB units. On figure 21 the test setup is defined. The test was conducted with the same method as laboratory exercise 5[5].

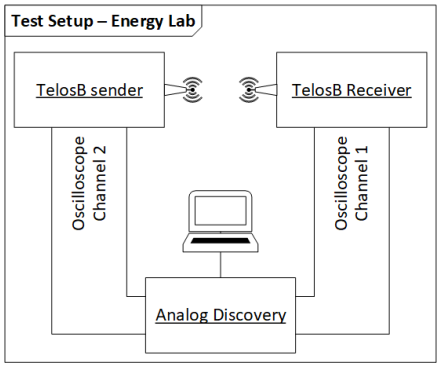

Figure 21: *Energy lab test setup.*

The measurements can be seen on figure 22, 23, 24 and 25.
Inspecting the values of the three signal strengths on Figure 22, 23, 24 and 25, and converting these voltages to a current, provides the results found in Table 2.
As can be 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

[5][Madsen(2018)]

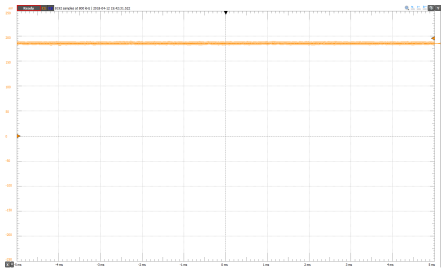

Figure 22: *Voltage drop at resistor $R = 10\Omega$ in series with Receiver (yellow), radio on, doing nothing.*

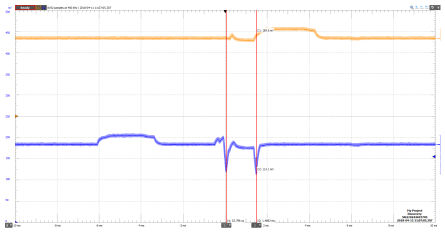

Figure 23: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at 0.0dBm.*

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.
As can be seen on Figure 26, turning on the radio is indeed expensive compared to having it turned off. A fast observation will result in a protocol decision that turns off the radio when not needed and on when needed. Of course 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 Energy Calculations, in which the data came from this section.

V. Test and performance

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

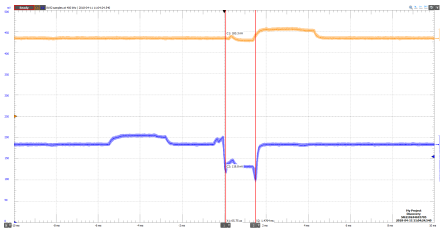

Figure 24: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at $-12.5dBm$.*

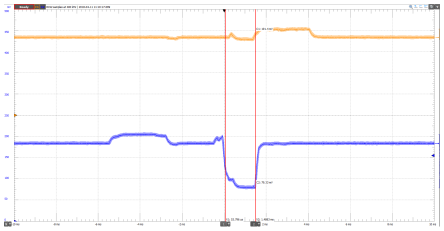

Figure 25: *Voltage drop at resistor $R = 10\Omega$ in series with Sender (blue) and Receiver (yellow), radio on, sending at $-25.0dBm$.*

VI. 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^{TM}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 III 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 43.5cm. 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 $1.087 * 10^3$ packets per round. But as our train is vastly slower than an actual runner, and the 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 XXX packets and that was not possible due to limits of the hardware, as seen in Figure POWERFIFG REF it takes 15ms to send a packet without accounting for other tasks going on at the telosb, eg. handling computations. We divided the packets per round by 4 $\frac{packetsPerRound}{4} = 271.852$ as a result needing $57.016ms$ per packet for the hardware to handle the speed. A picture of the setup can be seen in figure 27. For detailed calculations, please see appendix XXX.

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.*

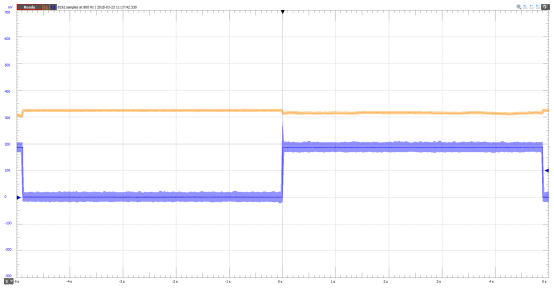

Figure 26: *Voltage drop at resistor $R = 10\Omega$ in series with TelosB unit (blue) and battery (yellow), radio switching between on and off.*

VII. WiFi conditions

In our test we used two different WiFi chanenls (4 and 11) and we found the best and the worst by using the WiFi Analyzer app from the Android app store[**?**]. As seen in figure 28, channel 11 is used by Aarhus University WiFi but channel 4 looks rather free, so we picked these two as seen in table 3.

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

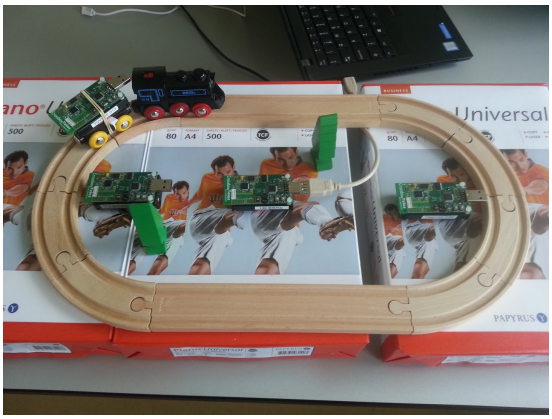

Figure 27: *A Brio toy train track with the base, north and south relay stations in the center and the runner put on top of the train.*

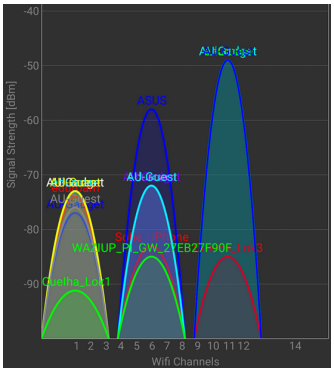

Figure 28: *WiFI on the test day. Channel 11 was busy and Channel 4 was a quiet channel*

VIII. Results

Using the test setup and mechanics described in section V, 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 hopping algorithm, the implemented stop-and-wait ARQ protocol and distance measurements were correct and working. All results are gathered from the base station.

To quickly examine 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 packet.

***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 shows the final result of each completed scenario lasting 48 minutes each, which equals about 144 rounds with the train. 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.

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.*

Sn.	p_1	p_2	p_3	p_4	p_5	p_6
1	29148	4066	3512	22169	34167	20700
2	29838	8115	8499	17880	23563	21111
3	29258	3706	4021	21944	32830	21244
4	30523	1232	8425	19137	28874	23188

Table 4: *Data from scenarios 1-4 completed with 144 runs.*

As expected, the results vary depending on the track length and the channel used. When using a length of 56 centimeters (scenario two and four), we see

$p5 =< p_1$, meaning the base station received the same or less data packets than requested. In scenario four they are even fairly close. If $p5 > p_1$, 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 56cm. Changing channels between 11 and 4 does not seem to increase the amount of data packets received or lower the number not acknowledged in time.

Figure 29 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.

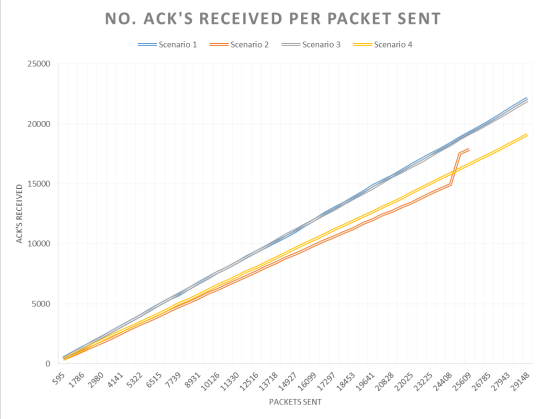

Figure 29: *No. ACK's received per packet sent.*

Figure 30 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, which means that changing channel from 11 to 4 does not have a impact on the received number of data packets when using length=43.5. However changing the length from 43.5cm to 56cm does. Scenario 4 looks to be the most successful one, as the number of data packets received is close to the number of packets sent. Scenario 1 and 3 sees more data packets being received than requested, a probable reason is multipath propagation.

Figure 31 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 packages, now that the runner node will

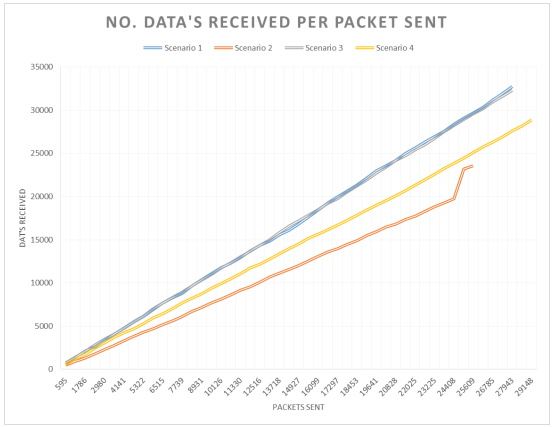

Figure 30: *No. DATA's received per packet sent.*

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.

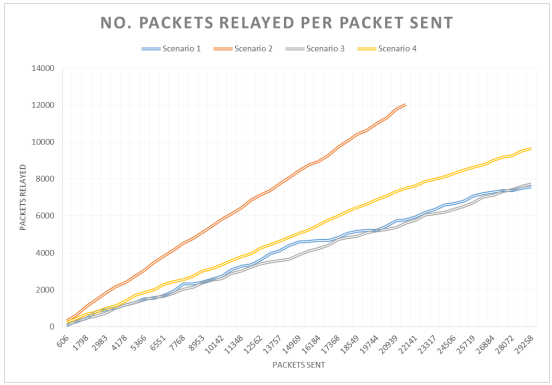

Figure 31: *No. packets relayed per packet sent.*

Figure 32 shows the number of packets not acknowledged over the data link in time per packet sent. All scenarios follows each-other, which points to a systematically error 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 33 and 34 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 34. A explanation may again be deep fading.

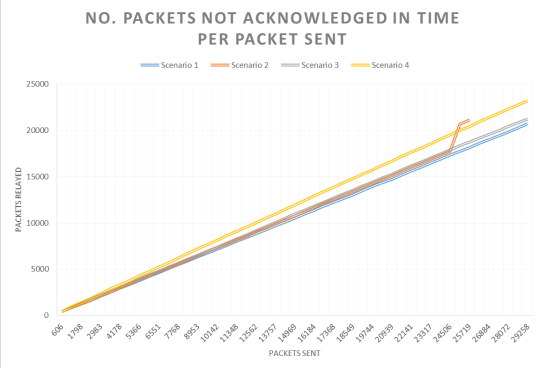

Figure 32: *No. packets not acknowledged in time.*

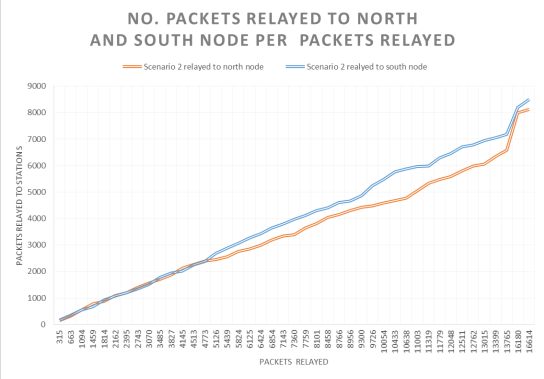

Figure 33: *No. packets relayed to north and south node in scenario 2.*

IX. Discussion

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

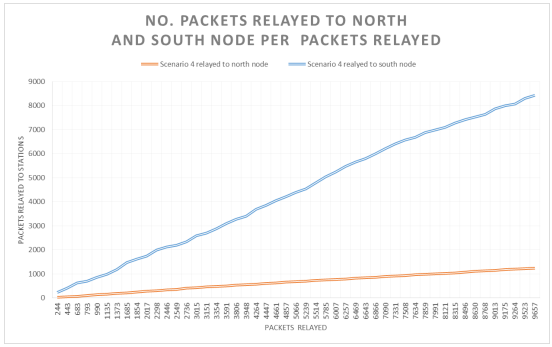

Figure 34: *No. packets relayed to north and south node in scenario 4.*

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 nodes able to relay between each other then that would open of the network to be scale of network. This would make it able to cover a even bigger track. Then we could also make some nodes to enter sleep mode because we would know the runner was on the other side of the track.

Our protocol in the current state is susceptible to attackers who takes advantage of the fact that our base station will keep using a relay if it can continue to give a good RSSI and heart rate data. This abuse would stop our wireless sensor network from operating correctly.

X. Conclusion

References

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

[Madsen(2018)] Jens Kargaard Madsen. Lab exercise - Energy & Power consumption. *Department of Engineering, AU*, pages 1–3, 2018.