

Demodulator Design for Collaborative Signal Reinforcement in Sensor Networks

Todd B. Fleming and Peter M. Athanas
Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
Email: {tbfleming, athanas}@vt.edu

Abstract

Nodes in sensor fields and in autonomous swarms of mobile robots need to communicate; this usually requires individual nodes to either consume a significant amount of energy, carefully position themselves, precisely align carrier phase, or use complex coding. We propose allowing multiple independent and mobile wireless transmitters to share the power-consumption burden by combining low-power transmissions into a unified higher-power transmission. This is not a distributed beamforming or space-time coding approach. Instead, all nodes simultaneously broadcast the same symbol on the same channel without using different code books and without aligning carriers. This paper briefly describes a modulation scheme (Random-Phase Frequency Shift Keying, RPFSK), a synchronization scheme (Ensemble), and a node selection scheme which make this possible. It then describes an RPFSK demodulator design in detail for Ensemble nodes. It also covers both baseband and over-the-air test results.

1. Introduction

Both sensor networks and swarms of autonomous mobile robots may perform useful tasks, such as measuring temperature and soil moisture levels for crop management, monitoring seismic activity for earthquake prediction, or even monitoring troop movements on a battlefield. Robot swarms may work cooperatively to find missing people or to sweep an area for mines. All of these activities require nodes to communicate with each other and with other devices. This usually requires individual nodes to either consume a large amount of energy, carefully position themselves, precisely align carrier phase, or use complex coding.

We propose allowing multiple independent and mobile wireless transmitters to share the power-consumption burden by combining low-power transmissions into a higher-power transmission. The approach described here is not a distributed beamforming or space-time coding approach. Instead, all nodes simultaneously broadcast the same symbol

on the same channel without using different code books and without aligning carriers. This approach is focused in three areas — modulation, synchronization, and repeater selection. There exist modulation schemes [1], [2] that have the property that when multiple nodes transmit the same symbol on the same channel at the same time, the modulated signals combine to produce a stronger signal without requiring carrier alignment. These modulation schemes are useful when combined with synchronization schemes [1] that enable multiple independent nodes to simultaneously transmit identical symbols. The third area of research examines how to decide which nodes should participate in a transmission [1].

This paper has the following organization. Section 2 briefly describes a modulation scheme (*Random-Phase Frequency Shift Keying*, RPFSK) that supports signal reinforcement without carrier alignment. Section 3 briefly describes a synchronization scheme (*Ensemble*). Section 4 describes a method for repeater selection. Section 5 describes a frame format that is compatible with the demodulator design that appears in the next section. Section 6 describes an RPFSK demodulator design in detail that is used in *Ensemble* repeaters. Section 7 covers the test environment and test results. Finally, Section 8 examines possible extensions to this work.

2. Random-Phase Frequency Shift Keying

There are several modulation schemes [1], [2] that satisfy the conditions noted in the previous section; this section examines one such scheme, *Random-Phase Frequency Shift Keying* (Fig. 1) [2]. Like *frequency shift keying*, each symbol is associated with a frequency. Transmitters split each symbol period into several chips and give each chip a different initial random phase. When the transmissions from several nodes combine over the air, some parts of each symbol reinforce each other and some parts weaken each other (Fig. 2). On average, if n nodes transmit the same symbol simultaneously then the combined signal's power will be n times as large as the power from a single

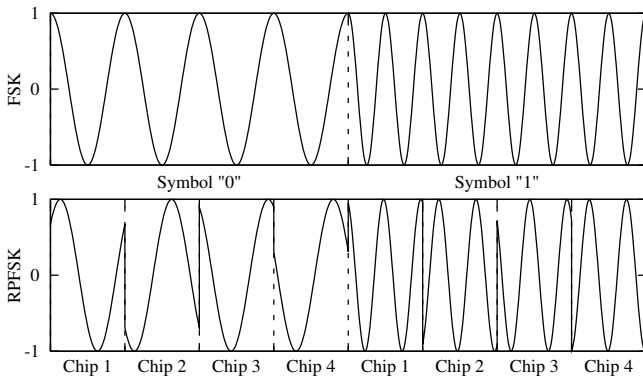


Figure 1. Comparison of FSK with RPFSK. RPFSK signals are divided into chips; each chip has a random initial phase.

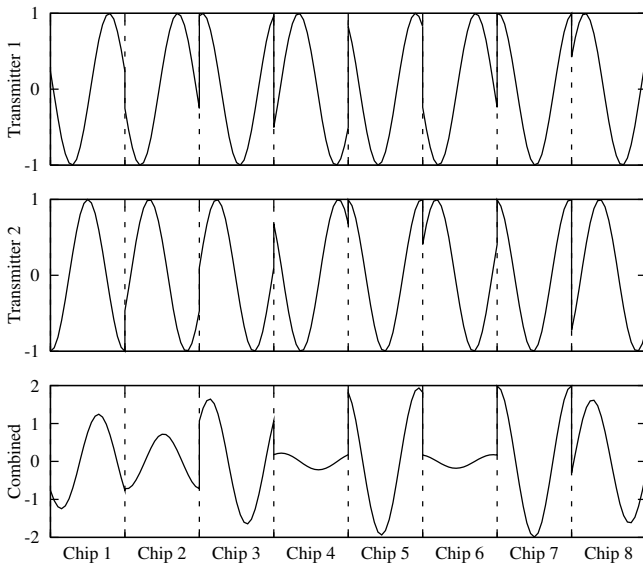


Figure 2. Combination of RPFSK signals from two transmitters. Each transmitter chooses a different initial phase for each chip. After combining, some chips are weakened and some are strengthened; on average, this produces a power gain that is equal to the number of transmitters.

transmitter [2]. The power gain's variance decreases as the number of chips increase. RPFSK has been tested over the air in the laboratory using two independent transceivers and one receiver. The two transceivers successfully produced a combined signal that was stronger than the individual signals; this combined signal had a lower bit error rate and a lower frame loss rate than the individual signals as measured by the receiver. This test used the over-the-air synchronization scheme described below.

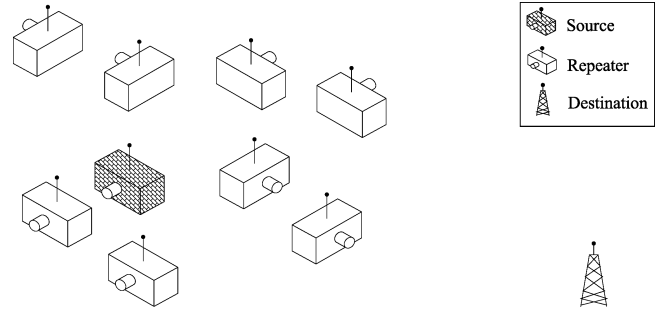


Figure 3. Ensemble node configuration. The source camera node (shaded) enlists the aid of the other nodes to reinforce the message that it needs to send to the destination.

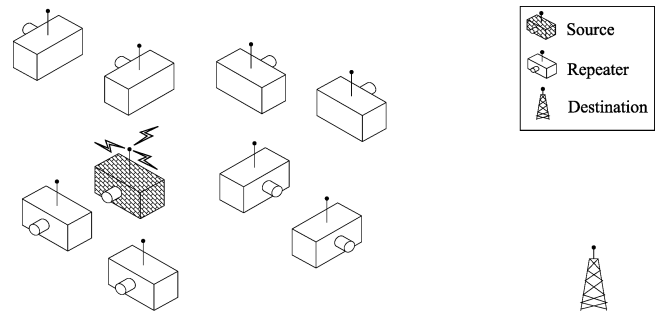


Figure 4. The source node broadcasts a symbol to the nearby nodes. Since they are nearby, the signal is strong and they decode it in a fraction of the time it would take if the signal was weak.

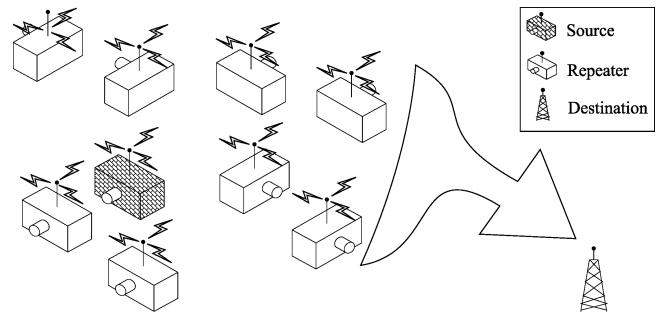


Figure 5. All nodes simultaneously broadcast the same symbol. The nodes use a modulation scheme that enables the combined signal to have a much larger combined power than the individual signals.

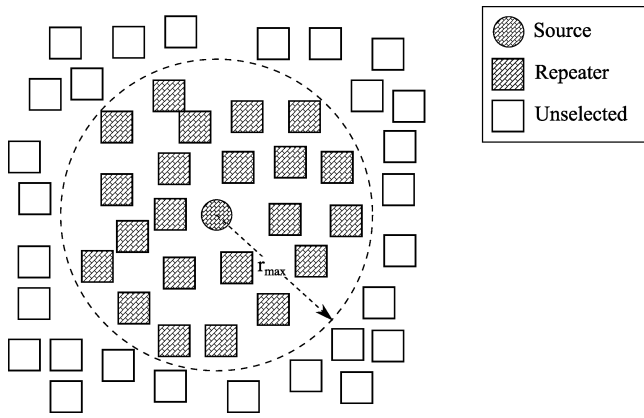


Figure 6. The source picks nodes to be repeaters.

3. Ensemble Synchronization

The synchronization scheme described here is named *Ensemble* [1]. When a node needs to transmit a message, it enlists the aid of its nearby neighbors. The source node then switches to transmit mode and the neighbors switch to repeater mode (Fig. 3). The source node transmits symbols sequentially (Fig. 4). The repeaters listen for activity on the channel. When they detect a symbol on the channel, they immediately start transmitting that same symbol for a fraction of the symbol period (Fig. 5). They remain silent for the rest of the symbol period then start monitoring the channel for further activity.

The duration that the repeaters send a symbol is based on their positions relative to the source node. The farther the repeaters are from the source node, the shorter the duration must be to prevent inter-symbol interference. When the repeaters are close to the source, the repeat duration is longer and the combined signal is stronger.

Ensemble synchronization is similar to the synchronization scheme described by Laneman *et al.* [3] for space-time-coded systems. Unlike [3], *Ensemble* does not rely on orthogonal channels; instead, the repeaters all transmit on the same channel as the source without using different codes. *Ensemble* receivers do not need to know about the presence of *Ensemble* repeaters in order to decode messages; the combined signal has the same modulation and coding as the original signal. *Ensemble* does not require precise channel estimation or node placement; only the distance between the source and farthest repeater must be known. Finally, *Ensemble* works on modulation schemes that provide power gain, not just diversity gain.

4. Repeater Selection

A source node needs to pick nearby nodes to be repeaters in order to send a message to the destination (Fig. 6). The most direct way to select nodes to be repeaters is to select

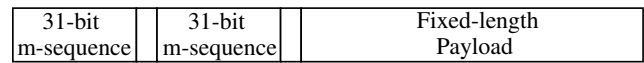


Figure 7. *Ensemble*/RPFSK frame format. Repeaters lock onto the first 31-bit m-sequence and the destination locks onto the second. Each m-sequence is followed by a 1-bit pad.

them by their distance to the source. One possible procedure is as follows: Start with an empty set. If the current set of nodes will not produce a signal strong enough to reach the destination, then add the nearest unselected node to the set and repeat. Future work may consider other selection approaches. For example, the source might avoid picking nodes that have low batteries; this may extend the average node lifetime.

5. Frame Format

An *Ensemble* system built on RPFSK uses a frame format similar to Fig. 7. Each frame has two 31-bit m-sequence headers; repeaters lock onto the first header and the destination locks onto the second. Each header is followed by a padding bit to prevent the demodulators from losing data. The payload length, and thus the length of the whole frame, are fixed.

The RPFSK demodulator described below requires a 31-bit header (m-sequence) at the beginning of each frame. The demodulator uses this header to detect the beginning of valid data; the demodulator also uses this header to find the symbol boundaries of the incoming signal. A padding symbol follows the header to give the demodulator enough time to flush its pipeline when it switches from header detection mode to data acquisition mode; this is necessary because of the demodulator's deep pipeline.

An *Ensemble* repeater cannot repeat data until it has locked onto the header. Thus, repeaters can not reinforce the first m-sequence. Without reinforcement, the header will be too weak for the destination to detect. The frame has a second header to solve this problem; repeaters lock onto the first header then repeat the second one, which is now strong enough for the destination to lock onto. The two headers have different m-sequences so the repeaters and destination will lock onto the appropriate points in each frame.

6. RPFSK Demodulator

An RPFSK demodulator needs to perform two tasks in an *Ensemble* repeater. First, it must properly decode each incoming symbol in a fraction of the symbol period. Second, it must accurately determine the beginning of each symbol period so the repeater will not create inter-symbol interference. This section describes a demodulator which meets these goals.

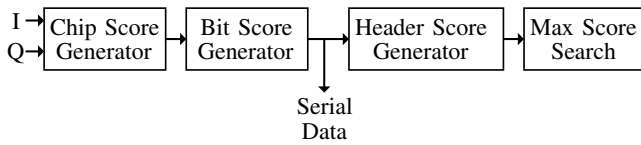


Figure 8. RPFPSK demodulator. The I and Q channels come from a quadrature downconverter.

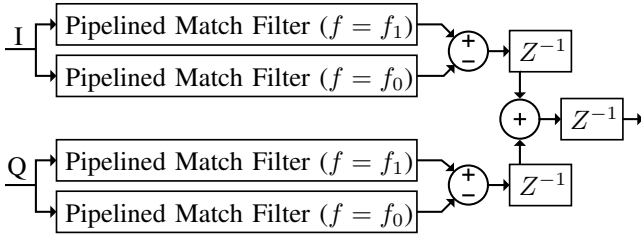


Figure 9. RPFPSK chip score generation. f_0 and f_1 are the frequencies for values 0 and 1 respectively. The score is valid five clock cycles after the end of a chip.

The demodulator uses four stages (Fig. 8) to complete its tasks. Two of these stages decode symbols and help align symbols; the remaining stages only align symbols. Recall that RPFPSK symbols are divided into chips; each chip contributes equally to a symbol's value. The demodulator's first stage examines each chip and generates a score; this score indicates both the likely value of the chip and the strength of the match. The second stage combines one or more chip scores from the first stage to produce bit scores. Like chip scores, each bit score gives both the bit's value and an indication of how good the match is. The second stage's output decodes to a valid symbol only after the demodulator has determined the proper alignment. Prior to that, the bit score is only used by the remaining stages. Both the first and second stages generate scores for every sample so that every possible alignment is covered.

The third stage compares bit scores to a 31-bit m-sequence to generate a header score. The header score indicates how close a group of 31 demodulated bits align with the received signal; a non-0 header score also marks the beginning of a frame. The fourth stage searches for the largest header score; the largest score indicates the best header and symbol alignment. The demodulator starts decoding symbols after the fourth stage found the best header alignment.

The following subsections describe each stage in detail.

6.1. Chip Score Generator

The first stage (Fig. 9) of the demodulator uses match filters to generate a chip score every clock cycle. Each filter (Fig 10) compares the incoming signal with a reference frequency (f) and generates a positive number that

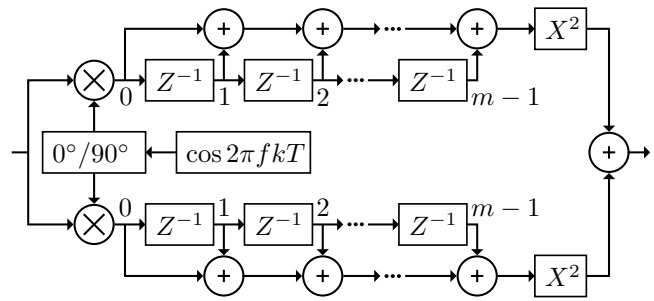


Figure 10. Phase-independent match filter. This detects the presence of a certain frequency component (f) in the last m -sample chip received. It produces a result at the end of a chip.

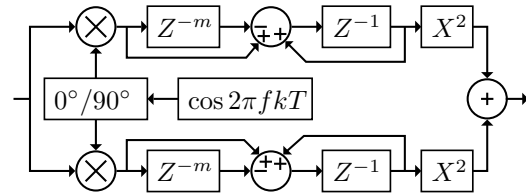


Figure 11. Simplified match filter. This filter is equivalent to Fig. 10, except it uses fewer adders and produces a result one clock cycle later.

indicates the strength of the match. Fig. 11 and Fig. 12 are refinements of this filter; Fig. 11 reduces the number of adders and Fig. 12 increases the pipeline depth to help meet timing constraints. Four of these match filters are combined in Fig. 9 to produce a chip score. If the resulting score is positive, then the chip is likely a 1; if it is negative, then the chip is likely a 0. The magnitude of this score indicates differential signal strength; larger magnitudes indicate a stronger decision than smaller magnitudes do. Sequential chip scores feed into the bit score generator.

6.2. Bit Score Generator

The second stage generates bit scores from chip scores (Fig. 13). A bit score is simply the sum of the scores of the chips that make up a symbol. The circuit in Fig. 13

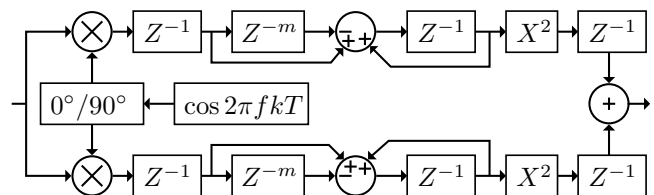


Figure 12. Pipelined match filter. This filter has additional delays after its multiplier stages to help meet timing constraints.

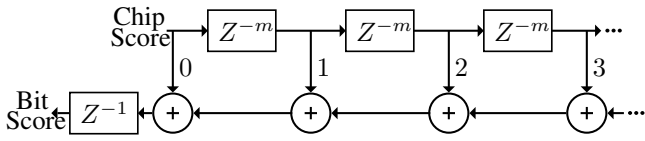


Figure 13. This combines the scores from several m -sample chips to produce a bit score. The number of stages matches the number of chips to analyze; the score is valid six clock cycles after the end of the most recent chip was received.

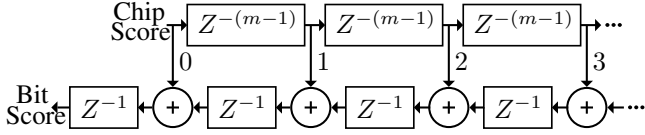


Figure 14. This is equivalent to Fig 13, but is more likely to meet timing.

uses delays to prevent chip scores from overlapping each other; every sample in a bit contributes to exactly one constituent chip score. Like chip scores, a large positive bit score indicates a strong 1 while a large negative bit score indicates a strong 0. Figure 14 is equivalent to Fig. 13, but improves timing.

6.3. Header Score Generator

The third demodulator stage (Fig 15) compares the scores from the most recent 31 bits to an m-sequence to produce a header score. This stage requires a perfect match; the score is 0 if any decoded bit does not match the pattern. The magnitude of the score indicates alignment precision; good symbol alignment produces larger header scores than poor ones do. Figure 17 is equivalent to Fig 15, but is more likely to meet timing constraints. The next stage uses header scores to find the best alignment.

6.4. Maximum Header Score

The fourth and final stage (Fig. 18) searches for the maximum header alignment score at the beginning of a

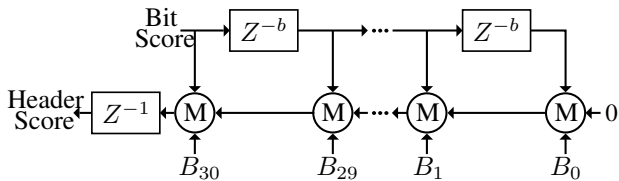


Figure 15. This circuit compares the 31 most-recent bit scores with a 31-bit m-sequence to produce a header score. Each incoming symbol is b samples long; the m-sequence pattern is B_i ; the match function (M) is described in Fig 16.

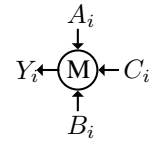


Figure 16. Match function used in Fig 15. If the sign of the bit score (A_i) matches the pattern (B_i) and all lower-order bits match (indicated by a non-0 C_i), then the resulting score (Y_i) is the sum of C_i and the absolute value of A_i . If A_i or any lower-order bits do not match, then the result is 0.

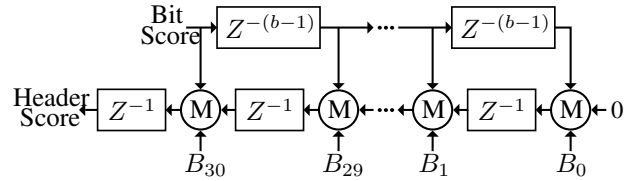


Figure 17. This is equivalent to Fig 15; the rearranged delays help the implementation meet timing constraints.

frame. It does this by calculating the maximum value of the most recent 2^p header scores and comparing them to the score that was generated b cycles ago, where b is the number of samples in one symbol and p is the smallest integer such that $2^p > b$. The best header alignment is found when this score is the same as the maximum score and is not zero. The demodulator switches to decoding mode once this best match is found. The following describes how the fourth stage calculates the maximum value of the most recent 2^p header scores using only p two-input maximum functions.

Let $\{x(0), x(-T), x(-2T), \dots\}$ be the sequence of header scores produced by Fig. 17. We abuse notation by using $Z^{-i}x$ to represent $x(-iT)$. The maximum value (y_p) of the most recent 2^p values of this sequence is given by

$$y_p = \max(Z^0x, Z^{-1}x, Z^{-2}x, Z^{-3}x, \dots, Z^{-(2^p-1)}x). \quad (1)$$

This can be rearranged to

$$y_p = \max \left(\begin{array}{l} \max(Z^0x, \dots, Z^{-(2^p-1)}x), \\ \max(Z^{-2^{p-1}}x, \dots, Z^{-(2^p-1)}x) \end{array} \right) \quad (2)$$

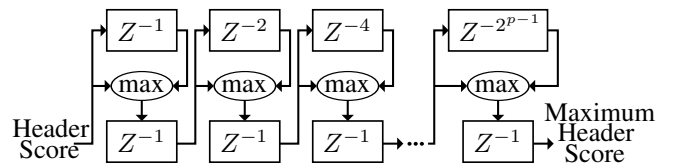


Figure 18. This calculates the maximum value of the most recent 2^p header scores. The result is valid p cycles after the last header score was received.

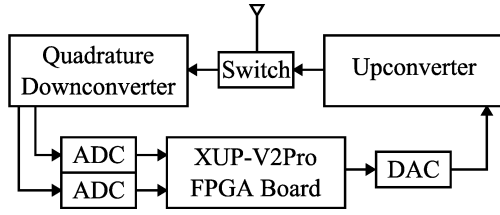


Figure 19. Block diagram of radio platform used for testing.

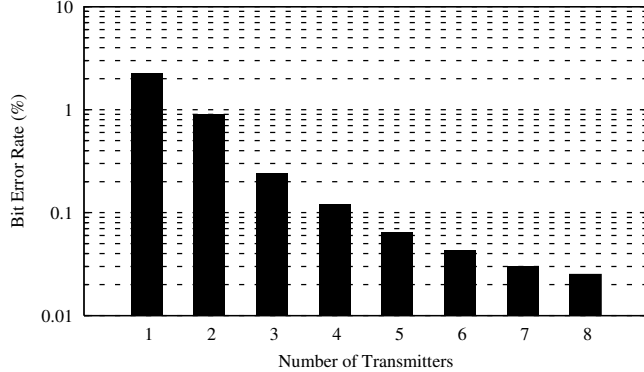


Figure 20. The bit error rate drops as the number of RPFSK transmitters increase.

then factored to

$$y_p = \max \left(\begin{array}{c} \max \left(Z^0 x, \dots, Z^{-(2^{p-1}-1)} x \right), \\ Z^{-2^{p-1}} \max \left(Z^0 x, \dots, Z^{-(2^{p-1}-1)} x \right) \end{array} \right). \quad (3)$$

Substitution produces

$$y_p = \max \left(y_{p-1}, Z^{-2^{p-1}} y_{p-1} \right). \quad (4)$$

Equation (4), when recursively applied to itself, describes how a circuit may calculate the maximum value of the most recent 2^p header scores using delays and p two-input maximum functions. However, this circuit will not meet timing requirements; the p two-input maximum functions form a tree with a propagation delay that's likely longer than a clock cycle. Equation (5) is similar to (4), but has clocked delays inserted into the tree. The circuit produced by this equation is in Fig. 18.

$$\begin{aligned} Z^{-p} y_p &= Z^{-p} \max \left(y_{p-1}, Z^{-2^{p-1}} y_{p-1} \right) \\ &= Z^{-1} \max \left(Z^{-(p-1)} y_{p-1}, Z^{-2^{p-1}} \left(Z^{-(p-1)} y_{p-1} \right) \right) \end{aligned} \quad (5)$$

7. Testing

The investigators tested this approach using a custom radio platform (Fig. 19) built on the Xilinx XUP-V2Pro FPGA (Field-Programmable Gate Array) board. The FPGA

contains the baseband modulation and demodulation logic. It drives the modulated signal into a 14-bit Digital to Analog Converter (DAC) running at 100 MSPS (Mega Samples Per Second). The DAC drives a double-sideband upconverter; the transmitted signal has a bandwidth less than 100 MHz centered at 2.05 GHz. A quadrature downconverter feeds I and Q channels into two ADCs (Analog to Digital Converters) for reception; the ADCs feed the FPGA and have the same sample rate and resolution that the DAC has. Both the upconverter and downconverter connect to a single antenna through a digitally-controlled switch. This platform tests both RPFSK power reinforcement at baseband and RPFSK power reinforcement with *Ensemble Synchronization* over the air.

A single node tested RPFSK power reinforcement at baseband. For this test, the FPGA contained eight RPFSK modulators, a noise generator, and a demodulator. The FPGA synchronized the modulators with each other internally and combined the signals with noise. A PC controlled the number of active modulators and monitored the demodulator's output to measure the packet drop rate and the bit error rate of the combined signal. The DAC, ADCs, and analog components were not used in this mode. This experiment tested 10000 packets for each run; each run had a different number of modulators active and all runs had the same background noise level. Both the bit error rate (Fig. 20) and the packet loss rate (Fig. 21) fell as the number of modulators increased. This demonstrated that RPFSK signals reinforce each other at baseband.

Three nodes were used to test RPFSK and *Ensemble Synchronization* over the air. The source, repeater, and destination nodes were arranged in laboratory experiments so that neither the source nor the repeater were close enough to the destination for the destination to receive a message from either. However, the destination was close enough to receive a reinforced message. The source transmitted messages over the air using RPFSK. The repeater reinforced each symbol using RPFSK and *Ensemble*. The destination successfully received messages.

8. Future Work

It is possible to extend the concepts presented in this paper with additional modulation schemes, synchronization schemes, node selection schemes, and with partitioning and multiple-access schemes. In addition, it should be possible to explore the theoretical implications of these concepts including bandwidth utilization, individual and global power utilization, obtainable data rates, and obtainable bit error rates.

As noted earlier, RPFSK has properties that are promising for collaborative reinforcement. RPFSK deserves further investigation, but there may be other collaborative modulation strategies that overcome the limitations of RPFSK. For

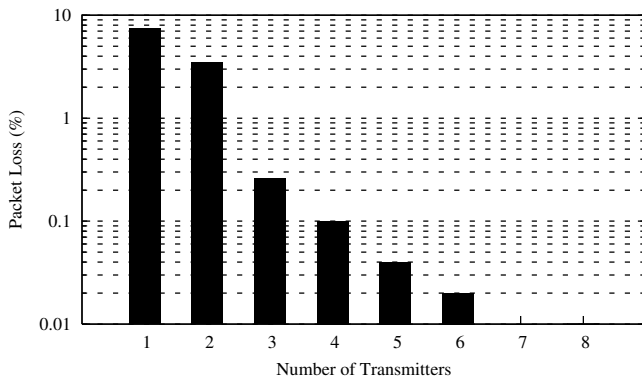


Figure 21. The packet loss rate drops as the number of RPFSK transmitters increase. No packets are lost when 7 or 8 modulators are used

example, this paper does not address multiple access; the current channel models can only handle a single message at a time. This paper also does not address partitions; large networks may need subdivisions to make complex node selection schemes tractable. We anticipate that the various schemes will each have a different set of advantages and disadvantages, thus it seems prudent to explore the schemes' theoretical implications in order to compare them with one another.

These schemes need a sound theoretical foundation to aid evaluation. Bandwidth utilization is one aspect of this work that needs investigation. For example, we have not characterized how bandwidth-limiting filters may effect signal reinforcement, data rate, and bit error rate. Power consumption is another aspect. For example, node selection algorithms effect both individual and global power consumption. Finally, data rate and bit error rate needs to be analyzed for each scheme. The entire theory will provide a set of tradeoffs that help implementors choose among the various schemes and choose values for parameters of those schemes.

Acknowledgment

The authors thank Tingting Meng and Mark Jones. Both inspired many of the concepts presented here and were active in their development.

This work was supported by the Office of Naval Research under contract N00014-05-1-0179.

References

- [1] T. Fleming and P. Athanas, "Collaborative synchronization for signal reinforcement in sensor networks," in *Proc. IEEE 21st Int. Conf. Advanced Information Networking and Applications*, to be published.
- [2] T. Meng and P. Athanas, "Collaborative signal reinforcement in sensor networks," in *Proc. IEEE 21st Int. Conf. Advanced Information Networking and Applications*, to be published.
- [3] J. N. Laneman and G. W. Wornell, "Energy-efficient antenna sharing and relaying for wireless networks," in *Proc. IEEE Wireless Communications and Networking Conf.*, vol. 1, Chicago, IL, Sep. 2000, pp. 7–12.