Nanonetwork Minimum Energy coding

Muhammad Agus Zainuddin, Eugen Dedu, Julien Bourgeois
Institut FEMTO-ST, DISC department
Numérica, Cours Leprince-Ringuet, BP 21126, 25201 Montbéliard, France
E-mail: {FirstName.LastName}@femto-st.fr

***Abstract*—Nanotechnology is generally considered a technology of the future. It promises to have many implications in various fields, and create revolutionary methods in some circumstances. Due to their size, nanodevices have limited capacities in terms of energy, computation and transmission among others. Networking them allows to increase their effectiveness, and also their communication range. However, data transmission consumes power, which is very precious in such devices. As such, communication between nanodevices in the Terahertz band have been investigated using low-power Time Spread-On Off Keying (TS-OOK) modulation. A characteristic of this modulation is that energy is required only for transmitting bit 1, since bit 0 is "transmitted" as silence (no energy). We exploit this property in the Nanonetwork Minimum Energy coding we propose in this paper. This coding reduces the number of 1s in data transmitted by source by encoding more often used symbols with fewer 1s. As such, it yields energy efficiency, but also reduces absorption noise and interference between devices, and increases information capacity. Results of this algorithm with various types of real files show notable improvements. It is able to reduce the energy up to 100%, depending on probabilities of 0 and 1 in input data.**

***Index Terms*—nanonetwork; energy consumption; Huffman coding.**

I. Introduction

Nanodevices are devices between one and several hundreds nanometers. Due to their size, they have limited capacities in terms of computation, energy and transmission range. Connecting them through a wireless network, so that they can exchange information, increases their usefulness [1]. Nanosensors are nanodevices which do sensing functions at nano scale, e.g. detect chemical compounds in concentrations as low as one part per billion, or the presence of virus and bacteria [2]. They need to regularly send data they sense to other devices, so their communication is an important parameter to consider. Wireless nanosensor networks will enable for example advanced applications in health monitoring [3], detect virus or harmful bacteria and then destroy it [4], multimedia communications, and surveillance systems against Nuclear, Biological and Chemical (NBC) attacks at nano scale [5].

Currently, there are mainly two alternatives for communication in nanonetworks. In molecular communication, sender encodes information in molecules and release them in the environment, and receiver decodes the information upon their reception [6]. The second type is the classical electromagnetic communication, used in this paper. It uses TS-OOK modulation in Terahertz band (0.1–10THz), which allows transmission rates up to several Terabits per seconds [2]. In TS-OOK modulation, bits are sent at very regular intervals, and 1 is transmitted as a pulse and 0 as silence (no transmission). Therefore, transmitting 0 instead of 1 increases energy efficiency, but also reduces absorption noise and interference between devices [7]. Reducing the absorption noise would increase the information capacity for single user case, while reducing the interference would increase the information capacity for multi user case.

In this paper, we introduce a source coding algorithm for nanosensor networks whose goal is to encode the data to be transmitted in order to increase the number of bits 0 in detriment to 1s. Data is divided in fixed size symbols. In a dictionary, more often used input symbols are mapped to output symbols with fewer bits 1. Before transmitting data, sender replaces input symbols with output symbols, and the receiver replaces received symbols back to original symbols. In the process, sender takes also into account the distance between 1s in output symbols. Depending on input data, bit 1 reduction can go up to 100%. In this paper we also analyse its energy efficiency and transmission robustness using several types of real files.

The paper is organised as follows. Sec. II presents related works. Sec. III presents Nanonetwork Minimum Energy (NME) coding algorithm and evaluation metrics. Sec. IV presents numerical results of the performance of NME coding using several types of file, both in energy efficiency and transmission robustness. The paper is summarized in Sec. V.

II. Related works

Compression techniques are usually used to reduce the redundancy in the information. A classical compression technique is Huffman coding, where the most often used symbols are encoded with fewer bits [8]. Our proposed method is a variation of Huffman coding. In both algorithms, symbols are ordered according to their frequency on input data. However, whereas in original Huffman algorithm the more frequent symbols have fewer bits, in our method the more frequent symbols have the same number of bits, but fewer number of 1s.

There are many variations of Huffman code. Abrahams [9] gives a comprehensive list. She considers fixed-to-variable, and variable-to-fixed source coding. Our method is fixed-to-fixed. Input data can be infinite, can have lexicographic constraints (Hu-Tucker problem), the codeword length can

This work has been funded by the Ministry of Education and Culture, Indonesia (Ph.D. grant no. 435/E4.4/K/2013).

have constraints, coding can have unequal cost code symbols (Karp problem). Our method is similar to the latter variant, Karp problem. Whereas in classical problems the cost of a symbol is the number of its bits, in Karp problem the cost of a symbol depends on its bit *values*, i.e. the cost of symbol i is $c(0)M(i) + c(1)N(i)$, with M the number of bits 0 in symbol i, N the number of bits 1, and $c(0)$ and/or $c(1)$ different than 1.

More specifically, there have been some works in energy efficient coding to reduce the frequency of occurrences of bits 1 that can be used in wireless networks. Erin et al. [10] proposed Minimum Energy (ME) coding to transmit more frequent symbols using fewer bits 1 in wireless communication. Our method is similar, with some differences due to characteristics of nanosensor networks. For example, codewords with the same number of bits 1 are not ordered in Erin's method, whereas in our method they are ordered according to the distance between 1s. This is because in nanosensor networks it is preferred to have greater distance between adjacent 1s, for two reasons. First, to have more relaxed constraints for energy harvesting, since a larger distance between adjacent 1s gives the sensor more time to harvest the energy for the next pulse transmission [11]. Second, to have a more relaxed (less activity) channel with respect to absorption noise in high traffic; for example, it is better to transmit 1010 sequence instead of 1100, because in the first case when sending the second bit the noise from the first bit transmission still exists in the channel. Erin's article is more theoretical, whereas ours presents numerical results with different values of symbol size on various types of input data.

Prakash et al. [12] propose another approach by encoding m bits into n bits ($n < n$) with weight 1. This method does not need statistics about input data, but requires more bandwidth since the size of output is bigger than input. Chi et al. [13] extend Prakash's method by using larger m and n, and provide more numerical proofs.

Chi et al. [14] use variable length code with minimum average code weight. To the best of our knowledge this method is prone to error. When an error happens, the subsequent symbols will be decoded with errors even if there is no error in the transmission after the error.

Kim et al. [15] propose a Modified Minimum Energy (MME) coding where codewords are divided in several subframes and a bit is added in front of each subframe: 1 means no high bits in the subframe, and 0 means at least one high bit in the subframe. The purpose of subframes is to allow the receiver to sleep for the duration of subframes with only 0s, thus reducing energy on receiver. This gives good results only in specific types of data, where adding a bit of data still decreases the energy; moreover, it increases data size.

ME and MME codings are analysed in the context of Coded Division Multiple Access (CDMA) wireless sensor networks (WSN) [16]. Contrary to this, our paper proposes a variant of ME which takes into account some characteristics of wireless nanonetworks.

III. Nanonetwork minimum energy coding

A. NME algorithm

Nanonetworks based on electromagnetic communication using TS-OOK modulation in Terahertz band have large bandwidth. The limitation in such networks are energy, computational complexity, and transmission range. Networking nanosensors in multi-hop fashion allows to reduce the computational complexity and to increase the transmission range. In TS-OOK modulation, 1 bits are transmitted with a femtosecond-long Gaussian pulse, with total energy 0.1 aJ [17], while 0s are transmitted as silence (no transmitted signal).

In this paper, we propose Nanonetwork Minimum Energy coding to reduce the energy usage for communication between nanosensors. It is a simple algorithm, suitable to the small power available in nanosensors. Data is transmitted from sender to receiver(s) as bits 0 and 1, and received as bits 0 and 1. The idea is to transmit the most often used blocks of bits with fewer 1s, in order to decrease the energy used to send the data. The algorithm for nanosensor networks is the following:

1) Segmentize the binary input sequence into blocks (symbols) of n bits.
2) Create a table of used symbols and their frequency.
3) Create another table by sorting the symbols in decreasing order of their occurrence level, and then encode more often used symbols with fewer 1s. Output symbols with the same weight are sorted in decreasing order of the largest distance between consecutive 1s in the output symbol.

If the order of the output symbols with the same weight is not taken into account, NME coding is the same as ME coding. For example, the available 4-bit symbols with 2 bits 1 are the following: 0011, 0101, 0110, 1001, 1010, 1100. ME coding orders them in ascending order, like previously written. Instead, NME orders them in descending order of the distance among the bits 1: 1001, 1010, 0101, 0110, 1100, 0011. Thus, more often used symbols are encoded with more spaces between 1s, which is more suitable to nanonetworks, as stated before.

Note that the output of NME algorithm has the same number of total bits as the input. The only difference is the number of 1s, the output of NME algorithm having less number of 1s than the input.

In the following, we will detail the algorithm. In step 1, the binary input sequences are segmented into blocks of n bits, afterwards the binary sequence is converted into symbols of $A = a_1, a_2, \ldots, a_N$. A denotes the set of possible output from the random variable X. The probability mass function is denoted by $P_i = P(X = a_i)$ for $i = 1, 2, ..., N$ (where $N = 2^n$). In practice, not all the symbols are used in transmission process. In this case, the set of used symbols can be defined as $Au = a_1, a_2, \ldots, a_M$, where $M \leq N$.

In step 2, the table of used symbols and their frequency (number of occurrences) is created, as shown in Table I. This

Symbol	Frequency
$a_i(1)$	$n(1)$
$a_i(2)$	$n(2)$
$\vdots$	$\vdots$
$a_i(M)$	$n(M)$

TABLE I
STEP 2 IN NME CODING.

Input symbols A_i	Frequency	Output symbols A_o
$a_i(1)$	$n(1)$	$a_o(1)$
$a_i(2)$	$n(2)$	$a_o(2)$
$\vdots$	$\vdots$	$\vdots$
$a_i(M)$	$n(M)$	$a_o(M)$

TABLE II
STEP 3 IN NME CODING.

table allows to count the number of 1 bits. The total weight (the number of 1s) for original data is:

$$N_{original} = \sum_{i=1}^{M} n(i) \times w(a_i) \quad (1)$$

where $n(i)$ is the number of occurrences of symbol i, and $w(a_i)$ is the Hamming weight of symbol i (the number of 1s in symbol i) [18].

In step 3, a new table (the dictionary) is created from the previous table by sorting the symbols based on their frequency of occurrence, as shown in Table II. More often used symbols appear upper in this table, i.e. $n(i) \geq n(j)$ for $i < j$. The dictionary is created before the transmission and it does not change afterwards. The total weight of NME output is:

$$N_{NME} = \sum_{i=0}^{M} n(i).w(a_o(i)) \quad (2)$$

B. NME properties

1) Definitions: An n-bit symbol is a string of n bits. A mapping is a correspondence table between a symbol and its encoded symbol, so a function $f : S_n \rightarrow S_n$, where S_n is the set of all n-bit symbols. A mapping is injective.

The following is an example of a 2-bit mapping:

$$00 \rightarrow 01$$
$$01 \rightarrow 11$$
$$10 \rightarrow 10$$
$$11 \rightarrow 00$$

It has four symbols of 2 bits each, both in input and output.

2) Number of mappings computing:

Theorem. The number of possible n-bit mappings is $2^n!$.

Examples. There are $2! = 2$ possible 1-bit mappings, $4! = 24$ possible 2-bit mappings, and $8! = 40320$ possible 3-bit mappings.

Proof. An n-bit mapping is on the form:

$$0...00 \rightarrow b_{11}b_{12}...b_{1n}$$
$$0...01 \rightarrow b_{21}b_{22}...b_{2n}$$
$$0...10 \rightarrow b_{31}b_{32}...b_{3n}$$
$$...$$
$$1...11 \rightarrow b_{N1}b_{N2}...b_{Nn}$$

The mappings are generated by the various arrangements of the N symbols, which yields $P_N = N!$ possible mappings. Looking at the number of lines, N is the total number of symbols of n bits, so $N = 2^n$. So the number of n-bit mappings is $N! = 2^n!$.

3) The best $2n$-bit mapping is equal or better than the best n-bit mapping:

Theorem. The set of n-bit mappings is a subset of $2n$-bit mappings set $\Rightarrow$ The best $2n$-bit mapping is equal or better than the best n-bit mapping.

Proof. Suppose the following generic n-bit mapping:

$$0...00 \rightarrow b_{11}b_{12}...b_{1n}$$
$$0...01 \rightarrow b_{21}b_{22}...b_{2n}$$
$$0...10 \rightarrow b_{31}b_{32}...b_{3n}$$
$$...$$
$$1...11 \rightarrow b_{N1}b_{N2}...b_{Nn}$$

We build the following $2n$-bit mapping:

$$0...00 \rightarrow b_{11}b_{12}...b_{1n}b_{11}b_{12}...b_{1n}$$
$$0...01 \rightarrow b_{11}b_{12}...b_{1n}b_{21}b_{22}...b_{2n}$$
$$0...10 \rightarrow b_{11}b_{12}...b_{1n}b_{31}b_{32}...b_{3n}$$
$$...$$
$$1...11 \rightarrow b_{N1}b_{N2}...b_{Nn}b_{N1}b_{N2}...b_{Nn}$$

where each encoded symbol T of a symbol S is formed by concatenation of the two encoded symbols of n bits corresponding to the first and the second half of the symbol S. For example, if the 2-bit mapping contains $01 \rightarrow 11$ and $10 \rightarrow 01$, we will use $0110 \rightarrow 1101$ in the 4-bit mapping.

This $2n$-bit mapping transforms each symbol of $2n$ bits in the same manner as the original n-bit mapping. So this $2n$-bit mapping is identical to the original n-bit mapping. (Two mappings are identical if the encoded text is the same, except perhaps the last bits of the text.) This means that any n-bit mapping can be written as a $2n$-bit mapping. Q.e.d.

4) Comparison between the best n-bit mapping and the best $n+1$-bit mapping: An n-bit mapping could be better or worse than a $n + 1$-bit mapping. Such examples are provided later, in results section. For example, in Table V, NME 2-bit has a greater energy efficiency than 3-bit, whereas in Table VI it is the contrary.

5) Dictionary length: In the general case, the dictionary (coding table) too should be transmitted to receiver, before the data. So the dictionary length is an important parameter to consider. However, as the data size increases, the dictionary length becomes less and less important compared to data.

The biggest dictionary for an n-bit mapping contains all the possible n-bit symbols, i.e. 2^n symbols. Each symbol has n bits. Therefore, the dictionary contains $n2^n$ bits for input, and $n2^n$ bits for output, which gives $2n2^n$ bits. However, the dictionary can be sorted by the output symbols, so that only

the input be transmitted. So the Maximum Dictionary Length is $MDL = n2^n$.

In practice, not all the 2^n symbols are found in the data, but only M, with $M \leq 2^n$. In this case, the Used Dictionary Length is $UDL = Mn$.

C. Metrics to evaluate NME

1) Energy efficiency: The first metric we use to measure the coding improvement is the energy efficiency ξ. This is the most critical parameter for nanosensor networks, since the battery capacity in nanosensors is very limited [13]. During a transmission, both sender and receiver consume energy. On the sender side, the energy to transmit symbol i is $E_i = w_i.E_p$, where E_p is energy of a pulse in TS-OOK modulation. Current nano-transceivers are able to transmit a pulse with total energy 0.1 aJ [17]. The total energy required for the transmission can be obtained by multiplying the number of transmitted pulse and the energy per pulse. The energy to transmit a symbol is equal to the weight of the symbol multiplied by energy per pulse. By taking into account the frequency of occurrences of each symbol, the total energy for uncoded data is:

$$E_{original} = \sum_{i=1}^{M} E_{a_i(i)} \times n(i) = N_{original} \times E_i \qquad (3)$$

where $E_{a_i(i)}$ is the energy of input symbol i, $n(i)$ is frequency of occurrences of symbol i, $N_{original}$ is total number of 1s in the transmitted data, and E_i is energy per pulse.

NME coding aims to reduce the number of 1s in uncoded data. After performing the coding, two components need to be transmitted: data and dictionary. Then the total energy after NME coding can be described by:

$$E_{NME} = \sum_{i=1}^{M} E_{a_o(i)} \times n(i) + \sum_{i=1}^{M} E_{a_i(i)} \qquad (4)$$

where M is number of used symbols, $E_{a_o(i)}$ is the energy to transmit output symbol i, n_i is its frequency, while $E_{a_i(i)}$ is the energy to transmit symbol i the dictionary.

The energy efficiency is given by the percentage of energy reduction using NME compared to the uncoded data:

$$\xi = \frac{E_{original} - E_{NME}}{E_{original}} \times 100\% \qquad (5)$$

A positive value of ξ means that NME algorithm effectively reduces the power consumption in transmission process. However, if dictionary size is important compared to data size, ξ could be negative (NME coding requires more power than uncoded transmission).

As it was already written, the receiver too consumes energy. We consider that it consumes the same power when receiving a bit 0 or 1. Thus the only difference between uncoded and NME transmissions is the transmission of the dictionary. Given that the dictionary length is normally much smaller than data size, and that the receiver consumes much less power than the sender (e.g. 10% of the power used by sender, mainly because there is no power amplifier at receiver), in this paper we only concentrate on the energy used by the source.

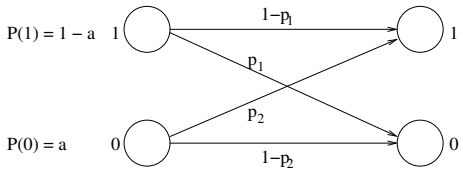

Fig. 1. Nanonetworks use the BAC channel model.

2) Robustness during transmission: Like in any wireless channel, wireless nanonetworks are prone to errors, with specific causes: molecular absorption, spreading loss, distances greater than e.g. tens of centimetres and so on. We use the Binary Asymmetric Channel (BAC) channel, shown in Figure 1, suitable to model nanonetwork losses [7]. In BAC model, 1 is correctly received with probability $1 - p_1$ and received with error with probability p_1. 0 is changed to 1 with probability p_2, with $p_2 \ll p_1$, because it corresponds to background noise. Input bit 0 comes with probability $P(0)$ and bit 1 with probability $P(1)$, with $P(0) + P(1) = 1$.

a) Codeword error rate: Since NME uses a dictionary table and the receiver encodes the symbol received back to the original symbol, a relevant metric for coding robustness is codeword (symbol) error rate (CER) of the channel, as opposed to individual bit error rate (BER) presented in Figure 1. CER computing is taken from [7]. The probability of bit error P_e can be calculated as follows:

$$P_e = p_2 P(0) + p_1 P(1) \qquad (6)$$

where the variables involved are the probabilities shown in Figure 1. A codeword in NME consists of n bits. If a bit in the codeword is wrong, the whole codeword is wrong. The probability of correct codeword (no bit error in the received codeword) is calculated by:

$$P_c = (1 - P_e)^n \qquad (7)$$

where n is the size of codeword. Assuming bit errors are not correlated, the CER is given by:

$$CER = 1 - P_c \qquad (8)$$

hence:

$$CER = 1 - (1 - p_2 P(0) - p_1 P(1))^n \qquad (9)$$

where n is codeword length and p variables from Figure 1.

b) Application to multimedia transmission: Until now we dealt with errors at channel level. Now we are interested in the upper level (NME) when such an error occurred. Since the decoding process in NME algorithm converts the output back into the input using the dictionary, the reconstructed symbol could be further distorted. For example, suppose that Table II contains the following two lines: 1010 $\rightarrow$ 0001 and 0101 $\rightarrow$ 1001. Suppose that sender wants to send message 1010. NME encodes it to 0001 before sending it through the channel. Due to a 1 bit transmission error, the message is received as 1001, and the receiver encodes it back to 0101. Summing up, symbol 1010 is received as 0101, which means that a 1 bit error during transmission produced a 4 bits error at receiver.

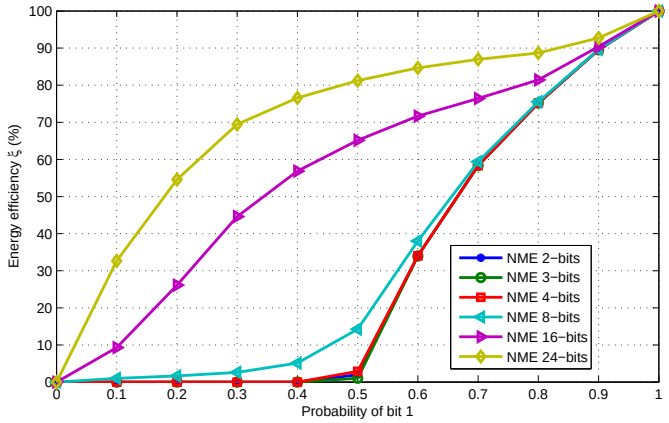

Fig. 2. Energy efficiency of NME coding for random data with various probabilities of bit 1.

In the following we use a practical example. As already presented in introduction, nanonetworks have applications in image transmission. The BAC channel create errors, so the received image could be different than the original one. To measure the quality of reconstructed image we use the classical Peak Signal to Noise Ratio (PSNR) metric. Suppose $I_i(x, y)$ is the input image and $I_o(x, y)$ is the received image. Then the distortion between input image and reconstructed image is:

$$e(x, y) = I_i(x, y) - I_o(x, y) \tag{10}$$

and the mean square error (MSE) is:

$$E_{ms} = \frac{1}{AB} \sum_{x=0}^{A-1} \sum_{y=0}^{B-1} e(x, y)^2 \tag{11}$$

where A and B are the image resolution on horizontal and vertical axis. Then PSNR is:

$$PSNR(dB) = 10 \log_{10} \left(\frac{255^2}{E_{ms}} \right) \tag{12}$$

The larger the PSNR value, the better the received image (closer to sent image) [19].

IV. Numerical results

We use MATLAB to obtain results.

A. Energy efficiency

We first generate random binary sequences of 10,000 bits with various frequencies of bit 1. Using equation (5), the energy efficiency of NME on data alone (when dictionary is known by receiver) is shown in Figure 2. The main result of the figure is that for a fixed input sequence size, the greater n, the greater the improvement, as expected. Also, when the input sequence has only 0 bits, the output is also all 0, so there is no coding improvement (0%); as the probability of 1 increases, the symbols with larger weight occur more often, and the coding improvement increases.

To evaluate NME more realistically, we do tests with several representative types of files: compressed and raw video/image, and program file. For instance, video files are bigger than image files; compressed video/image files have very few redundancy (this is the purpose of compression), so the number of bits 0 and 1 is about 50% each; a program file has many bytes 0. The particular files used in each category (`news_cif`, `bus_qcif` etc.) were chosen at random, without any specific reason. The following results present the energy used to send the data (the number of 1s in the output of NME), the dictionary (the number of 1s in dictionary) and their sum. The energy efficiency for NME coding is measured using equation (5).

Compressed video: `news_cif.mp4`. Transmitted bits: 7,692,136 bits = 0.92 MB. The number of 1s for original is 3,763,743 bits. The NME performance is shown in Table III. The largest improvement is achieved using NME 8 bit with 2.58% energy efficiency. As a side note, NME 24 bit reduces the number of 1s in coding data, but requires a large dictionary, which generates a negative energy efficiency.

Uncompressed video: `bus_qcif.yuv`. Transmitted bits: 11,556,864 bits = 1.38 MB. The number of 1s for original is 5,607,698 bits. The NME performance is shown in Table IV. NME gains in all cases, and the largest improvement is achieved using NME 16 bit with 53.81% energy efficiency.

Uncompressed image: `lena.bmp`. Transmitted bits: 532,912 bits = 65.1 kB. The number of 1s for original is 260,762 bits. The NME performance is shown in Table V. The largest improvement is achieved using NME 8 bit with 23.56% energy efficiency.

Compressed image: `lena.jpg`. Transmitted bits: 272,016 bits = 33.2 kB. The number of 1s for original is 132,740 bits. The NME performance is shown in Table VI. The largest improvement is achieved using NME 8 bit with 6.60% energy efficiency. Note that a negative energy efficiency appears for 16 bits, because the dictionary size (26.7 kB) is comparable to data size (33.2 kB).

Program file: `AdobeUpdater.dll`. Transmitted bits: 4,019,712 bits = 0.49 MB. The number of 1s for original is 1,680,819 bits. The NME performance is shown in Table VII. NME gains in all cases, and the largest improvement is achieved using NME 16 bit with 38.54% energy efficiency.

B. Robustness during transmission

The BAC channel error probabilities are set to $p_1 = 0.1$ and $p_2 = 0.004$ (i.e. p_2 is 25 times smaller than p_1).

c) Codeword error rate: Based on equation (9), the codeword error rate for various values of p_1, n and $P(1)$ is shown in Figure 3 (in each case, $p_2 = p_1/25$). For example, when $P(1) = 0.3$ (i.e. 30% of input bits are 1, and 70% are 0) and codeword has 8 bits, the values $p_1 = 0.1$ (probability of receiving 0 when sending 1) and $p_2 = 0.004$ (probability of receiving 1 when sending 0) yield 23% erroneous symbols. As expected, at fixed $1 \rightarrow 0$ error probability (a vertical line in the figure), the larger the n, the larger the codeword error rate, so the more vulnerable to error during transmission.

d) Application to multimedia transmission: Figure 4 presents the robustness during image transmission with various

Coding	Number of 1s in dictionary (bits)	Number of 1s in data (bits)	Number of 1s in total (bits)	Energy efficiency (%)	Dictionary length (byte)	Max dictionary length (byte)
Original	–	–	3,763,743	–	–	–
NME 2 bit	4	3,735,368	3,735,372	0.76	1	1
NME 3 bit	12	3,716,347	3,716,359	1.26	3	3
NME 4 bit	32	3,708,997	3,709,029	1.45	8	8
NME 8 bit	1,024	3,665,543	3,666,567	2.58	0.25 k	0.25 k
NME 16 bit	523,358	3,389,503	3,912,861	−3.96	127.8 k	128 k
NME 24 bit	3,708,769	1,961,620	5,670,389	−50.66	923 k	48 M

TABLE III
NME PERFORMANCE FOR NEWS_CIF.MP4 FILE (0.92 MB).

Coding	Number of 1s in dictionary (bits)	Number of 1s in data (bits)	Number of 1s in total (bits)	Energy efficiency (%)	Dictionary length (byte)	Max dictionary length (byte)
Original	–	–	5,607,698	–	–	–
NME 2 bit	4	5,569,261	5,569,265	0.69	1	1
NME 3 bit	12	5,392,470	5,392,482	3.84	3	3
NME 4 bit	32	4,428,079	4,428,111	21.04	8	8
NME 8 bit	1,024	3,326,281	3,327,305	40.67	0.25 k	0.25 k
NME 16 bit	271,466	2,372,978	2,590,444	53.81	54.3 k	128 k
NME 24 bit	1,980,761	1,891,442	3,872,203	30.95	0.5 M	48 M

TABLE IV
NME PERFORMANCE FOR BUS_QCIF.YUV FILE (1.38 MB).

Coding	Number of 1s in dictionary (bits)	Number of 1s in data (bits)	Number of 1s in total (bits)	Energy efficiency (%)	Dictionary length (byte)	Max dictionary length (byte)
Original	–	–	260,762	–	–	–
NME 2 bit	4	245,266	245,270	5.94	1	1
NME 3 bit	12	252,038	252,050	3.34	3	3
NME 4 bit	32	223,466	223,498	14.29	8	8
NME 8 bit	1,024	198,315	199,339	23.56	0.25 k	0.25 k
NME 16 bit	76,974	131,903	208,877	19.90	19.3 k	128 k
NME 24 bit	229,518	89,270	318,788	−22.25	57.3 k	48 M

TABLE V
NME PERFORMANCE FOR LENA.BMP FILE (65.1 KB).

Coding	Number of 1s in dictionary (bits)	Number of 1s in data (bits)	Number of 1s in total (bits)	Energy efficiency (%)	Dictionary length (byte)	Max dictionary length (byte)
Original	–	–	132,740	–	–	–
NME 2 bit	4	132,386	132,390	0.26	1	1
NME 3 bit	12	132,294	132,306	0.33	3	3
NME 4 bit	32	130,010	130,042	2.03	8	8
NME 8 bit	1,024	122,955	123,979	6.60	0.25 k	0.25 k
NME 16 bit	108,405	82,463	190,868	−43.79	26.7 k	128 k
NME 24 bit	132,011	42,469	174,480	−31.44	33 k	48 M

TABLE VI
NME PERFORMANCE FOR LENA.JPG FILE (33.2 KB).

Coding	Number of 1s in dictionary (bits)	Number of 1s in data (bits)	Number of 1s in total (bits)	Energy efficiency (%)	Dictionary length (byte)	Max dictionary length (byte)
Original	–	–	1,680,819	–	–	–
NME 2 bit	4	1,518,325	1,518,329	9.67	1	1
NME 3 bit	12	1,544,009	1,544,021	8.14	3	3
NME 4 bit	32	1,382,547	1,382,579	17.74	8	8
NME 8 bit	1,024	1,093,100	1,094,124	34.91	0.25 k	0.25 k
NME 16 bit	212,215	820,857	1,033,072	38.54	53.7 k	128 k
NME 24 bit	726,974	596,478	1,323,452	21.26	0.19 M	48 M

TABLE VII
NME PERFORMANCE FOR ADOBEUPDATER.DLL FILE (0.49 MB).

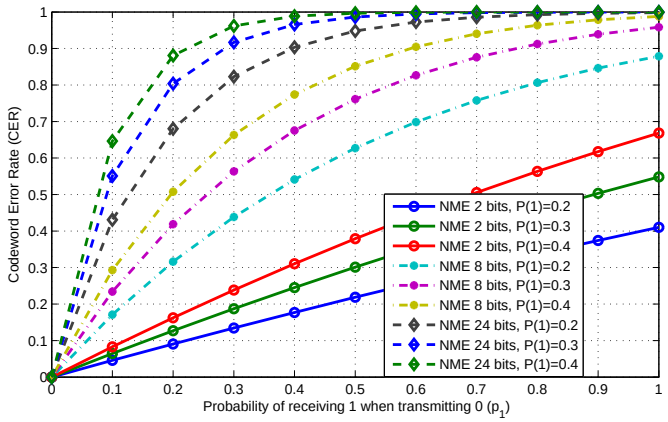

Fig. 3. Codeword error rate for various parameters.

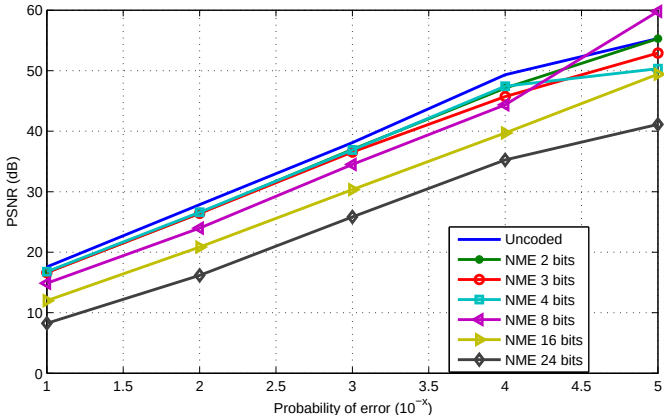

Fig. 4. PSNR for an image transmission.

probabilities of error. It can be seen that the bigger the number of bits in NME coding, the greater the efficiency, but the less the robustness/reliability. This is because, as explained in section III-C2, the error can increase when using a dictionary like NME does. This can be spotted easily in Figure 5, which presents the received image using various values for NME coding when transmitting the image through the channel.

V. Conclusions and Future Research

Communication between nanodevices using TS-OOK modulation requires energy to transmit 1 bits and no energy to transmit 0 bits. Since nanosensors transmit much data and have very limited energy, energy requirements of communication is very important. Based on this idea, in this paper we propose a method to reduce the number of 1s in data transmitted, by encoding more often used symbols using fewer 1s. We evaluate it with real files and investigate the code robustness during transmission. Numerical results show that the proposed algorithm saves energy depending on input data distribution, in some of our tests more than 50%, and in theory up to 100%. Results also show that our method is more vulnerable to channel errors, therefore it needs to be combined with error correction code.

Perspective research include increasing energy efficiency through an adaptive coding, where the encoding adapts in real-time to the data being transmitted, and investigating interference in multi user communication.

Acknowledgement

Authors are grateful to Josep Miquel Jornet for his comments on this article.

References

[1] I. F. Akyildiz, F. Brunetti, and C. Blazquez, "Nanonetworks: a new communication paradigm," *Computer Networks*, vol. 52, no. 12, pp. 2260–2279, 2008.

[2] J. Jornet and I. Akyildiz, "Channel modeling and capacity analysis for electromagnetic wireless nanonetworks in the terahertz band," *IEEE Transactions on Wireless Communications*, vol. 10, no. 10, pp. 3211–3221, Oct. 2011.

[3] G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, "Nano-sim: Simulating electromagnetic-based nanonetworks in the network simulator 3," in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, ser. SimuTools '13, ICST, Brussels, Belgium, 2013, pp. 203–210.

[4] I. F. Akyildiz and J. M. Jornet, "Electromagnetic wireless nanosensor networks," *Nano Communication Networks*, vol. 1, no. 1, pp. 3–19, Mar. 2010.

[5] J. M. Jornet and I. F. Akyildiz, "The internet of multimedia nano-things," *Nano Communication Networks*, vol. 3, no. 4, pp. 242–251, Dec. 2012.

[6] T. Nakano, M. J. Moore, F. Wei, A. V. Vasilakos, and J. Shuai, "Molecular communication and networking: Opportunities and challenges," *IEEE Transactions on NanoBioscience*, vol. 11, no. 2, pp. 135–148, Jun. 2012.

[7] J. M. Jornet, "Low-weight error-prevention codes for electromagnetic nanoneworks in the terahertz band," *Nano Communications Networks*, vol. 5, no. 1–2, pp. 35–44, mar-jun 2014.

[8] J. G. Proakis, *Digital Communications*, 4th ed. McGraw-Hill International, 2001.

[9] J. Abrahams, "Code and parse trees for lossless source encoding," in *Compression and Complexity of Sequences*, ser. 1997. Salerno, Italy: IEEE, Jun. 1997, pp. 145–171.

[10] C. Erin and H. Asada, "Energy optimal codes for wireless communications," in *IEEE Proceedings of the 38th Conference on Decision and Control*, Dec. 1999.

[11] P. Wang, J. M. Jornet, M. A. Malik, N. Akkari, and I. F. Akyildiz, "Energy and spectrum-aware mac protocol for perpetual wireless nanosensor networks in the terahertz band," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2541–2555, 2013.

[12] Y. Prakash and S. Gupta, "Energy efficient source coding and modulation for wireless applications," in *IEEE WCNC*, 2003.

[13] K. Chi, Y. Zhu, X. Jiang, and X. Tian, "Optimal coding for transmission energy minimization in wireless nanosensor networks," *Nano Communication Networks*, vol. 4, pp. 120–130, 2013.

[14] K. Chi, Y. Zhu, X. Jiang, X. Tian, and V. Leung, "Energy-efficient prefix-free codes for wireless nano-sensor networks using ook modulation," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 2670–2682, May 2014.

[15] J. Kim and J. G. Andrews, "An energy efficient source coding and modulation scheme for wireless sensor networks," in *IEEE 6th Workshop on Signal Processing Advances in Wireless Communications*, ser. 6. New York, USA: IEEE, Jun. 2005, pp. 710–714.

[16] C. Fischione, K. H. Johansson, A. Sangiovanni-Vincentelli, and B. Z. Ares, "Minimum energy coding in CDMA wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 2, pp. 985–994, Feb. 2009.

[17] J. M. Jornet and I. Akyildiz, "Femtosecond-long pulse-based modulation for terahertz band communication in nanonetworks," *IEEE Transactions on Communications*, vol. 62, no. 5, pp. 1742–1754, May 2014.

[18] B. Skalar, *Digital Communications, Fundamentals and Applications*. Pearson Education Asia, 2001.

[19] Y. Q. Shi and H. Sun, *Image and Video Compression for Multimedia Engineering, Fundamentals, Algorithms, and Standards*. CRC Press, 2007.

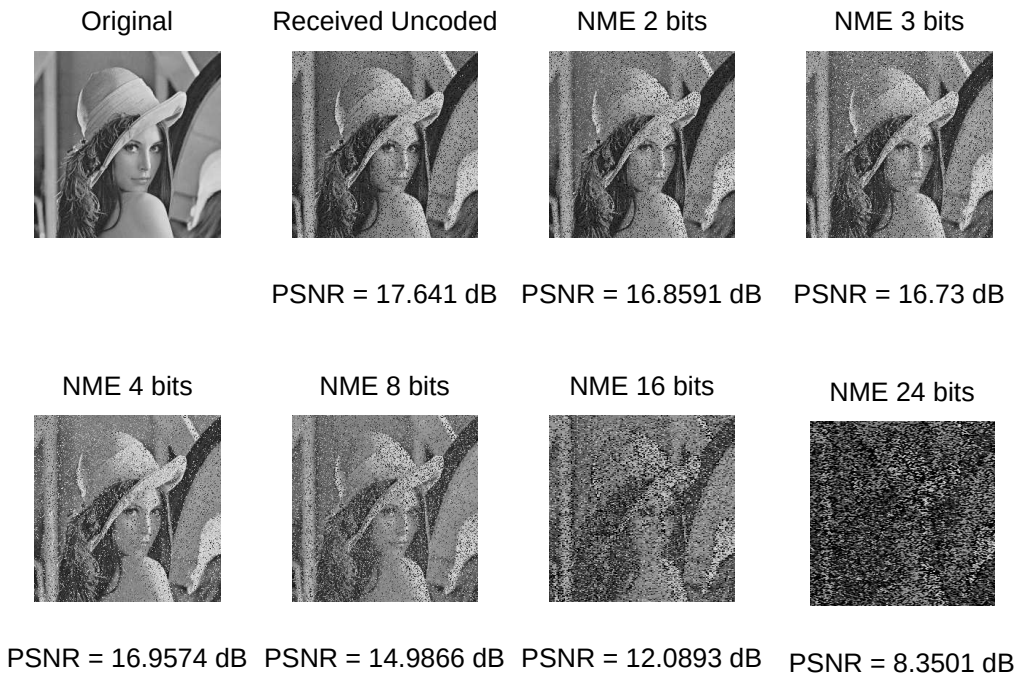

Fig. 5. Received `lena.bmp` image when transmitted through a BAC channel with $p_1 = 0.1$ and $p_2 = 0.004$.