Long-term Audio Observation by Wireless Sensor Networks with Filtering Strategies

NING-HAN LIU, CHENG-YI LI, SHU-JU HSIEH
CHENG-FA TSAI AND MIN-HUA SHAO

Department of Management Information Systems, National Pingtung University of Science & Technology, Taiwan, R.O.C.
E-mail: gregliu@mail.npust.edu.tw
1, Hseuh Fu Road, Nei Pu, Pingtung, 912, Taiwan, R.O.C.
Tel: 886-8-7703202; Fax: 886-8-7740306


Due to the wireless transmission capability and ease of replaceability of wireless sensor networks (WSN) nodes, they can be deployed in harsh environments for monitoring environmental changes. However, a sensor network faces the challenge of limited residual energy, particularly in audio sensing where large amounts of data are communicated and drainage of sensor energy is significant. In order to conserve energy, it is thus desirable for sensor nodes to intelligently analyze and filter sensed data to be delivered to the server. Nevertheless, data processing on the computationally limited sensor node needs to be lightweight. In other words, the data processing technique needs to operate efficiently and accurately under memory and processing constraints. In this paper, two approaches are proposed to filter redundant sensed audio data in order to prolong WSN lifetime. The first approach analyzes the fundamental frequencies in the distribution of audio data to prune unobserved audio, while the second approach compares the amplitudes of multiple audio signals to filter out duplicated data. Our experimental results prove that through applying the proposed approaches for wildlife detection, the sensor network lifetime can be significantly prolonged.

Keywords: Wireless sensor networks, audio detection, audio processing, audio recognition, power conservation.
1 INTRODUCTION

Due to rapid advances in computer hardware and wireless network communication technology, wireless sensor networks (WSN) technology has become prominent. It has become widespread in several applications that involve the sensing of the earth’s movement, temperature, humidity [1][2]. The main reasons for the popularity of WSN technology include: (1) low cost: the cost of each sensor is required to be less than a dollar so that it can be replaced affordably; and (2) ability to self-configure: when a sensor node has been powered on, it can automatically connect with neighboring nodes to establish a network and enable sensed information to be routed back to the server, which diminishes the cost of infrastructure construction [3]. Nevertheless, the lifetime of a sensor node is limited as each sensor node is battery powered. This issue is exacerbated by the fact that the energy required for a transmission is thousands of times more than that to perform a machine cycle computation on a sensor node. It thus follows that sensor energy consumption is hugely determined upon the number of transmissions made and the amount of information transmitted [3] [4].

WSN has mainly been used for sensing light intensity, humidity, temperature and images [5]. In contrast, WSN has rarely been used for acoustic sensing, for example, in detecting environmental sound or the differences in sounds emitted by different animals. However, research in acoustic sensing has meant that the technology behind it has become more mature [6][7]. In the 1970s, two important techniques were proposed, namely dynamic time warping (DTW) [8] and the hidden Markov model (HMM) [8]. These are the two most commonly used techniques for audio signal recognition. However, due to the limited computational power of existing sensor nodes, it is infeasible to perform complex calculations present in techniques such as DTW and HMM on the sensor itself. In particular, the amount of audio information that has to be stored is substantial in comparison to light, humidity and temperature information. Thus, some tradeoffs were proposed in implementing signal-processing algorithms, e.g. the system in [9] distributed the signal data to other sensor nodes to be processed in a collaborative fashion.

In our work, we consider the scenario whereby the WSN is deployed in the wild to detect animal sounds. In this scenario, the server only serves to collect the audio information coming from placing the node closest to the acoustic source – i.e. the animal being observed (rather than from all other nodes that have collected similar information). As shown in Figure 1, the number labeled on the link arrows connecting the nodes refers to the number of transmissions made. The total number of transmissions made is twelve, implying that a lot of unnecessary data has been delivered.

Two approaches are proposed in this paper. The first approach involves filtering unnecessary audio data by utilizing the distribution of fundamental frequencies in sensing data. The second approach filters redundant audio data
at the nodes that route data transmissions (or hops). The experiments conducted demonstrate that these two approaches prolong the lifetime of WSN.

This paper is organized as follows. Section 2 describes the hardware platform used. Section 3 introduces the audio signal filtering method for the data sensing node. Section 4 presents the duplicate audio filtering method for the data transmission node. Section 5 details the experiments performed and lastly, Section 6 concludes this paper and outlines directions for future work.

2 HARDWARE PLATFORM

Early research in WSN started in UC Berkeley [10], in which physically small sensing devices known as Smart Dust were invented using Micro-Electro-Mechanical-Systems (MEMS) technology. Their primary use was in military battlefields but in recent years, due to increasing interest in WSN, both academic and business communities have used them for other purposes. WSN has also spurred interest among various big corporations such as Intel and Microsoft, who have already started research in sensor networks with various future plans.
The sensor hardware used in this research is the Imote2 with IPR2400 (developed by Crossbow, Intel and Microsoft). The Imote2 is a new generation mote platform with operational and memory capabilities beyond traditional mote platforms. The processor in Imote2 is the Intel PXA271, coupled with 32MB SDRAM and 32MB Flash memory. It has been selected for this research because of its operating and memory capabilities that are comparatively stronger than conventional sensor platforms, allowing it to process audio signals. Nevertheless, we note that the Imote2 by itself does not have audio sensing capability. To enable audio data collection, we have created a basic audio detection sensor board that provides audio signals to the Imote2 through the ADC input interface of the Imote2. Figure 2 shows the audio detection sensor board we have built for the Imote2.

As shown in Figure 1, the sensor-node detecting sound is termed the sensing node (represented as s-node) and the upper layer sensor node that transmits the audio signal is termed the transmitting node (denoted as t-node). Our work can be divided into two parts. The first part involves handling audio detection with filtering at the s-node and the second part involves the comparison with filtering of audio signals at the t-node.

3 AUDIO SIGNAL FILTERING ON S-NODE

As shown in Figure 3, before the WSN is set up, the identification component is created through the fundamental frequencies distribution of training data and installed on the s-node.

When animal sounds have been detected in the environment, the s-node begins audio recording and executes the *Acoustic Filtering Module*. In addition to the identification component, other elements of this module include: (1)
Noise Reduction: to remove unnecessary noise and preserve the sound that the system requires; (2) Endpoint Detection: to determine the sound segments that need to be analyzed; and (3) Feature Extraction: to obtain the necessary information for the identification module.

If the signal is identified to be a candidate sound by identification modules installed on the s-node, the audio signal is passed to the server through the t-node for further recognition. Otherwise, if the signal is not required by the system, it is discarded and not transmitted to the server. This in effect reduces the number of data transmissions in the network and prolongs network lifetime.

3.1 Feature extraction
In our work, the audio signal feature selected is the fundamental frequency detected by the sum of magnitude difference function (SMDF) [8]. The function is used due to the minimal number of calculations required. The calculation of the SMDF is shown in equation (1) where $n$ is the distance by which the audio frame is right shifted, $x(p)$ is the amplitude of each point in the frame, and $M$ is the length of the frame.

$$D_m(n) = \sum_{p=0}^{M-1} |x_m(p) - x_m(p+n)| \quad n = 0, 1, 2, \ldots$$

The purpose of using SMDF is to shift an audio frame right by $n$ units and the overlapping segment with the original frame being mutually cancelled. Fol-
Following this, the absolute value is extracted, the combined sum is calculated and this process is repeated $M$ times until the $M$ inner product value is obtained. The final result obtained is shown in Figure 4. This figure shows the initial position with the minimum value of the SMDF being zero. Using this result, a threshold value can be set (dashed line) and the lowest point (dots) can be discovered, which is lower than the threshold value but is not the zero starting point. Lastly, after the index values of these points are discovered, the smallest index value discovered is the fundamental frequency of the audio signal.

With respect to the sensor’s identification module, to reduce sensor computational overhead, this research statistically groups the fundamental frequencies obtained via SMDF and compares the difference in the group distribution to enable a simpler identification process.

A segment of detected sound is divided into dozens of frames and dozens of fundamental frequencies are obtained after applying SMDF. If these fundamental frequencies have been tabulated directly, it would be difficult to observe the differences between sounds. Therefore, the fundamental frequencies first need to be grouped in the manner shown in Figure 5. Our experimental trials maintain a group size of 7 (since this makes the variance of every frequency interval less than an acceptable value and results in the greatest difference among the frequency distributions of the animal species) in order to minimize the information-processing load and at the same time, to increase the difference between fundamental frequency groupings.

While using the same parameter settings, several sound samples from the animal species are statistically analyzed to extract each group’s mean count value. The number of the sampled audio signals for each animal species that comes from the subsidiary farm in our school is 100. The fundamental frequencies distribution obtained is shown in Table 1. The value in each cell of Table 1 shows the average number of the fundamental frequencies in a group in percentage corresponding to the frequency of the sound made by the animal. Furthermore, the variance of each cell is lower than 4. Observation of
the table with the variance constraint shows that the table is sufficient for animal recognition. After features have been extracted from the sound data and the modules have been trained, the parameter necessary for the identification array is obtained. This can be used as a comparison to the distribution of fundamental frequencies sensed by the sensor.

3.2 Audio signal detection on the s-node
It is common that only irrelevant background audio is captured when audio is sensed in external environments. Therefore, in order to capture sounds of interest in the environment efficiently, it is desirable to only initiate audio

\[ \text{FIGURE 5} \]
Fundamental frequencies distribution chart

\[ \text{TABLE 1} \]
Fundamental frequencies distribution for various animal sounds

<table>
<thead>
<tr>
<th>Sound</th>
<th>Frequency</th>
<th>&lt; 61</th>
<th>61~80</th>
<th>81~100</th>
<th>101~130</th>
<th>131~170</th>
<th>171~210</th>
<th>&gt; 210</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bull</td>
<td>28%</td>
<td>24%</td>
<td>20%</td>
<td>16%</td>
<td>0</td>
<td>8%</td>
<td>4%</td>
<td></td>
</tr>
<tr>
<td>Sheep</td>
<td>36%</td>
<td>8%</td>
<td>24%</td>
<td>12%</td>
<td>0</td>
<td>16%</td>
<td>4%</td>
<td></td>
</tr>
<tr>
<td>Dog</td>
<td>12%</td>
<td>16%</td>
<td>8%</td>
<td>28%</td>
<td>16%</td>
<td>12%</td>
<td>8%</td>
<td></td>
</tr>
<tr>
<td>Horse</td>
<td>20%</td>
<td>16%</td>
<td>24%</td>
<td>12%</td>
<td>4%</td>
<td>12%</td>
<td>12%</td>
<td></td>
</tr>
<tr>
<td>Chicken</td>
<td>28%</td>
<td>20%</td>
<td>8%</td>
<td>4%</td>
<td>24%</td>
<td>0</td>
<td>16%</td>
<td></td>
</tr>
<tr>
<td>Bullfrog</td>
<td>32%</td>
<td>12%</td>
<td>16%</td>
<td>0</td>
<td>20%</td>
<td>12%</td>
<td>8%</td>
<td></td>
</tr>
</tbody>
</table>
data collection when the current audio signal differs significantly from the previous audio signal sensed. An instance of this is when the volume of the audio signal suddenly increases. For this purpose, we propose a basic technique to determine if an audio signal has changed.

This technique primarily uses the first few audio frames as the background audio, as the basis for audio comparison. When the change in the magnitude of the audio volume exceeds a preset threshold, it implies that an interesting sound may be present and this sound can be recorded to be further processed. The proposed method is further detailed as follows:

Firstly, the average energy of the background frames and variance are calculated. In equation (2), $E_n$ represents the energy of the audio frame, $n$ represents the frame number that is preset, $N_b$ and $E_b^{\text{mean}}$ represent the magnitude and the mean energy of the background frames respectively. In equations (3) and (4), $\sigma_{E_b}^2$ is the variance of energy and $E_b^{up}$ is the upper bound of the energy in the background frames.

When the audio signal energy exceeds $E_b^{up}$, the interesting audio is recorded until the energy is lower than the lower bound and then transmitted to the server through the t-nodes if the audio signal data is not filtered out at the audio signal comparison stage.

$$E_b^{\text{mean}} = \frac{1}{N_b} \sum_{n=1}^{N_b} E_n$$  \hspace{1cm} (2)

$$\sigma_{E_b}^2 = \frac{1}{N_b} \sum_{n=1}^{N_b} (E_n - E_b^{\text{mean}})^2$$  \hspace{1cm} (3)

$$E_b^{up} = E_b^{\text{mean}} + \sigma_{E_b}$$  \hspace{1cm} (4)

When the origin and destination points of the sound data have been confirmed, features can then be extracted from the sound signal segment. The steps are the same as the steps described in the previous subsection, i.e. the whole sound data segment is divided into audio frames and SMDF is used to calculate each audio frame’s fundamental frequency.

### 3.3 Identification method on the s-node

When the sound data features have been extracted, the preliminary audio identification on the sensor node is instantiated through applying the fundamental frequency comparison technique, i.e. when the fundamental frequencies distribution shows sound data that is similar to the one of the observed animal, the sound data is identified as the observed animal sound. The computation of the fundamental frequencies distribution for the sensed data follows the same way of Table 1.
The dissimilarity degree between two fundamental frequencies distributions is defined by the following function:

\[
\text{Dis}_\text{sim}(X,Y) = \sum_{i=1}^{n} |x_i - y_i|
\]

(5)

Where \(X\), \(Y\) are the fundamental frequencies distributions of the sensed data and the sample data of the observed animal respectively. Here, \(x_i\) and \(y_i\) are the fundamental frequencies in percentage value for group \(i\) and \(n\) is the total number of fundamental frequencies groups.

Equation (5) only calculates the dissimilarity between the fundamental frequencies distribution of sensed data and the one of the observed animal. Therefore, a predefined threshold \(d\) is used to decide whether the sensed data is the observed animal sound. That is, if the value of \(\text{Dis}_\text{sim}(X,Y)\) is less than \(d\), then \(X\) belongs to \(Y\). This is the basic identification process carried out at the sensor node end.

4 DUPLICATE AUDIO FILTERING ON T-NODE

Audio comparison at the t-node is the key contribution of this research and which is also the most challenging issue addressed. We propose a comparison method suitable for use in WSN, one that determines whether the audio information received within a specified time period is similar.

4.1 Audio amplitude comparison method

We compute the standard deviation of the difference in amplitude between two audio signals as their dissimilarity degree. As an example, in Figure 6, the top figure represents the original audio signal, the middle figure represents the audio signal with lower volume and the bottom figure represents the result that represents the difference in amplitude between the two signals. If the standard deviation of this difference is lower than or equal to a specified threshold, the two audio signals compared are regarded as the same.

As shown in equation (6), \(S_x\) and \(S_y\) represent the amplitudes of audio signals \(x\) and \(y\) respectively and a new signal sequence \(S_d\) is the combined difference between the two audio signals. Lastly, \(\sigma_{sd}\) is the standard deviation of the values in the sequence \(S_d\). In equation (7), \(E_d^{\text{mean}}\) represents the mean value of \(S_d\) and \(N_d\) represents the magnitude of elements \((E_x)\) in \(S_d\). If \(\sigma_{sd}\) is lower than or equal to a threshold \(T\), it implies that the two audio signals are the similar to one another.

\[
S_d = S_x - S_y
\]
If only the volumes of two signals are different, the associated $\sigma_{sd}$ is equal to the difference of the standard deviations of the two signals. However, the sampling errors will make $\sigma_{sd}$ larger than the difference of the standard deviations. Therefore, the difference of the standard deviations should be enlarged carefully to the threshold. Equation (8) shows the calculation steps for the threshold $T$, where $\sigma_{sx}$ is the standard deviation of amplitude in audio signal $x$, $\sigma_{sy}$ is the standard deviation of amplitude in audio signal $y$, and $W$ is a parameter to adjust the value of the threshold. As the detected audio signal information can differ at any time, this threshold value changes according to the difference of the standard deviations of the two signals.

\[ T = (\sigma_{sx} - \sigma_{sy}) \times W \]  

Another similar parameter that can be used for audio signal difference calculation is the zero crossing rate [11]. If the zero crossing rate is used instead of the amplitude, the resultant effect should theoretically be the same. However, according to our research, although it uses the same concept, the zero cross-
ing rate introduces an error in the calculation. This is because the audio signal is typically encoded in 8 bits and when the signal amplitude is too small, it will be recorded as zero.

4.2 Adjustment of audio starting position

The setting of the starting position for audio signals is also important in the calculation of the difference between two audio signals. In particular, a small difference in the sampling points can significantly affect the results obtained. To address this issue, we adopt the end-point detection method as it is not easily affected by noise.

End-point detection [8] is a common technique for handling audio signals. Its primary function is to search for the starting and ending positions in an audio source.

Firstly, the audio information is converted into a sequence of audio frames, whereby the energy and zero crossing rate of each frame is calculated as \( E_x \) and \( Z_x \) respectively and \( x \) is the sequence number of the audio frame.

We extract the foremost portion of the audio frame as the noise region and by applying the energy and zero crossing rate distribution, the mean value and variance are calculated. Equations (9) and (10) are used to calculate the mean energy and zero crossing rate. Equations (11) and (12) are used to calculate the variance of the energy and zero crossing rate.

\[
E_b^{\text{mean}} = \frac{1}{N_b} \sum_{x=1}^{N_b} E_x
\]  
(9)

\[
Z_b^{\text{mean}} = \frac{1}{N_b} \sum_{x=1}^{N_b} Z_x
\]  
(10)

\[
\sigma_{Eb}^2 = \frac{1}{N_b} \sum_{x=1}^{N_b} (E_x - E_b^{\text{mean}})^2
\]  
(11)

\[
\sigma_{Zb}^2 = \frac{1}{N_b} \sum_{x=1}^{N_b} (Z_x - Z_b^{\text{mean}})^2
\]  
(12)

After the variance of the energy and zero crossing rate have been calculated, the standard deviation values from variances are used to set the energy thresholds \((T_{EL}, T_{EU})\) and zero crossing rate threshold \(T_Z\), as shown in equations (13), (14) and (15). \( \alpha_1, \alpha_2 \) and \( \alpha_3 \) are user-defined adjustment parameters.

\[
T_{EL} = E_b^{\text{mean}} + \alpha_1 \sigma_{Eb}
\]  
(13)
\[ T_{EU} = E_b^{mean} + \alpha_2 \sigma_{E_b}, \quad \alpha_1 < \alpha_2 \]  

\[ T_Z = Z_b^{mean} + \alpha_3 \sigma_{Z_b} \]  

The above thresholds are used to search for the first audio frame with energy threshold above \( T_{EL} \), and this is set as \( N_V \). If subsequent frames after \( N_V \) have higher energy than \( T_{EL} \) and the subsequent frames after these frames have energy above \( T_{EU} \), then \( N_V \) would be regarded as the candidate for the starting frame, else \( N_V \) is not the starting frame and the algorithm continually searches the other subsequent frames for \( N_V \).

When \( N_V \) has been discovered, the frame before \( N_V \) with zero crossing rate below \( T_Z \) is regarded as the starting point (denoted as \( N_0 \)). If there are no frames before \( N_V \) with zero crossing rate above \( T_Z \), \( N_V \) is treated as the starting point of the audio signal. From \( N_V \) onwards until the energy is lower than \( T_{EL} \), the frame is noted as the end point (denoted as \( N_E \)). As such, the segment in between \( N_0/N_V \) and \( N_E \) is the region of the audio to be captured. This is illustrated in Figure 7.

End-point detection uses audio frames as the unit of measurement. As such, it needs to be noted that it does not provide an accurate sampling posi-
tion but rather, an accurate position of the audio frame for starting point
detection. Therefore, the sampling position of a discovered audio frame
needs to be readjusted and compared in order to determine the most suitable
starting position. Equation (16) is the conversion function from the audio
frame to the sampling position. Here, $F_{\text{index}}$ represents the audio frame posi-
tion, $F_{\text{size}}$ represents the size of the audio frame, $O_{\text{size}}$ represents the overlap
size, and $S_{\text{index}}$ is the resulting sampling point after the conversion. Due to
the $S_{\text{index}}$ is an roughly estimated starting position, there is a need to shift an
audio signal to find the best position as the starting point such that the $\sigma_{sd}$ in
equation (7) is the smallest. As shown in equation (17) (extended from equa-
tion (7)), $S_{x}^{\text{index}}$ is the amplitude sequence of audio $x$ with the starting posi-
tion $\text{index}$, where $S_{y}^{\text{index'}}$ uses a similar definition. Moreover, ‘Shift’ represents
the position shifted of the amplitude sequence for the estimation of the real
starting point. Given that the size of an audio frame is set as $k$, the forward
and backward shifts are $k/2$ times in order to find the smallest $\sigma_{sd}$, as shown
in equation (17).

$$S_{\text{index}} = (F_{\text{index}} - 1) \times (F_{\text{size}} - O_{\text{size}}) + \left\lfloor F_{\text{size}} / 2 \right\rfloor$$

$$s_{sd}^2 = \arg\min_{\text{Shift}} \left( \frac{1}{N_d} \sum_{x=1}^{N_d} (E_x - E_d^{\text{mean}})^2 \right)$$

where $S_{d} = (S_{x}^{\text{index}} \pm \text{Shift}) - (S_{y}^{\text{index'}} \pm \text{Shift}), \text{shift} = 1,2,\ldots k/2$

### 4.3 Propagation delay problem

As the data transition and collision times for sensor nodes and the time to
process audio data are not predictable, the t-node may need to send the audio
data back to the server before other similar audio transmissions are made by
the other s-nodes. Therefore, we design a basic protocol to guarantee that the
t-node would get all the audio data that needs to be compared to one another
before the audio signal can be sent out.

Let us assume that the topology of WSN is a tree structure. In other words,
any sensing data that is necessary is sent back to the server through some
path. Our method is such that after each sensor has detected the candidate
audio data, the s-node sends a tag message to all t-nodes in the path to the
server. The tag includes the node $id$ and a random serial number to identify
the audio data that may be sent through the path in the future from the s-node.
The tag transmitting priority is set to be higher than that for other messages
to ensure that the parent nodes are aware that the audio has been detected by
the s-node. In particular, if the audio data is dropped by a t-node in the path,
the decision-making t-node sends another tag to cancel the previous identification tag. After a t-node has received the audio data with all serial numbers, the t-node would then compare the audio data.

5 EXPERIMENTAL RESULTS

This section discusses the experiments carried out to further our study of the proposed methods. The first experiment explores the filtering ability of the identification approach on the s-node. Two measurements, namely false positive and false negative have been defined to illustrate the performance, as shown in Table 2. As defined in equation (18), the false positive for the animal observed is the value given by dividing the number(not_observed_animal) (i.e. the amount of test sound samples that are not from the observed animal) by the number(identify_as_observed_animal) (i.e. the number of sound samples that are identified as the sound from the observed animal but which should not be treated as such).

\[
False\_positive = \frac{\text{number(identify\_as\_observed\_animal)}}{\text{number(not\_observed\_animal)}} \tag{18}
\]

\[
False\_negative = \frac{\text{number(identify\_as\_not\_observed\_animal)}}{\text{number(observed\_animal)}} \tag{19}
\]

Similarly, in equation (19), the false negative is given by dividing the value of number(observed_animal) (i.e. the amount of testing data that produced by the observed animal) by the number(identify_as_not_observed_animal) (i.e. the number of sound samples that are not treated as the observed animal sound).

As the identification procedure on the s-node consumes additional energy for the computing of fundamental frequencies, the next experiments compare the power consumptions between s-nodes that operate with and without the identification filtering mechanism. From our extensive experimentation (up to hundreds of tests), we discovered that the energy consumed in the use of the identification procedure in sensing is only 17.3% more than sensing with-

<table>
<thead>
<tr>
<th>(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>False positive 23.7</td>
</tr>
<tr>
<td>False negative 19.3</td>
</tr>
</tbody>
</table>

TABLE 2
Result of false positive/negative
out the use of the procedure. The result shows that the approach is very useful for sensing audio in environments that frequently do not contain the sounds of animals of interest. Furthermore, transmission cost is more than five times that of the filter computation cost.

In the following experiments, the filtering ability in the t-node is explored. Firstly, we perform experiments with the $W$ parameter setting used in equation (8). We discover that the most desirable result is obtained when the $W$ value is 1.1. The experiment for the t-node highlights the impact to energy consumption between using filtered and unfiltered audio signals with the audio data transmission. An unfiltered audio signal implies that when a node receives the audio information, this information is directly transmitted without audio comparison, whereas for a filtered audio signal, the t-node will filter duplicate audio content and transmit the resulting portion of the signal to the server. The experimental settings are as follows. Assume that every t-node receives 2 identical audio signals every minute, with both signals having different starting positions. Every signal is 4 seconds in length, each node is 10 meters apart, the range of an acoustic source is 30 meters, and experiments run for 2 hours. The experimental results obtained are illustrated in Table 3. We observe that a higher amount of energy is expended when the unfiltered method is used, compared to when the filtered method is used.

The last experiment evaluates the lifetime of the WSN, in an outdoor deployment (in the wild). The animal that needs to be observed is bullfrog. Two groups of WSNs are used simultaneously. One group uses the two filtering approaches in the s-node and the t-node while the other one does not. Each group consists of 10 sensors (i.e. Imote2) and the sensors of different groups are distributed at the same location. Due to the difficulty of experimenting in the wild, we have used the time of the last transmitted audio data in the server to estimate the lifetime of the WSN. Furthermore, after the recognition of a sound by a human, the average percentage of sound from a bullfrog is 18% of all the animal sounds.

From the obtained results of several experiments running under the previous settings, we have shown that the lifetime of the WSN applying the filtering approaches is prolonged by up to 4-5 times compared to the one without the filtering approaches. We note that the results are sound sensitive, i.e. if the frequency of sound from the observed animal is high, the reserve energy is

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Energy Consumption (three 1.5V batteries)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unfiltered</td>
<td>37%</td>
</tr>
<tr>
<td>Filtered End-point detection</td>
<td>17%</td>
</tr>
</tbody>
</table>

TABLE 3
Energy consumed using different strategies
low. On the contrary, if the sound of the observed animal is low, the reserve energy is high. Moreover, the average accuracy of the data collected by the unfiltered WSN is 14% (less than 18% due to noise resulting from the heavy wind). The average accuracy of our filtered WSN is 73.5%.

6 CONCLUSIONS

By using tools such as WSN, the process of extracting data from the wild has evolved from typical manual data collection to autonomous data collection. However, sensor node capabilities are limited and there is growing research interest in machine learning to effectively use resource-constrained sensors.

In this paper, we have investigated audio signal data collection in a WSN and proposed methods to filter unnecessary/duplicate audio signals in order to extend the lifetime of the sensor. The procedures of the proposed approaches are simple enough to be to run efficiently on the sensor node. According to the experimental results obtained, the proposed methods effectively reduce overall network energy consumption. Nevertheless, there remains areas that should be improved and which can be addressed in the future. For instance, developing a technique to increase the accuracy of audio recognition through utilizing lightweight versions of more complicated methods (e.g. HMM and GMM). Moreover, other similarity metrics (e.g. cosine distance between two vectors; cross-correlation; histogram intersections) that measure the different degree between two signal sequences should be used and compared to each other if the mote has more memory and a more powerful processor in the future. Since the higher signal-noise ratio may cause a lower performance with our method, the noise irrelevant method will be our next research issue. Furthermore, another issue to tackle is to reduce the energy consumption of data transmissions by utilizing audio compression.

7 ACKNOWLEDGMENT

This work was supported by NSC of Taiwan under the contact numbers NSC97-2218-E-020-002 and NSC98-2200-E-020-004.

REFERENCES


