DOI: 10.1109/INFOCOM.2015.7218370

ExPerio - **Ex**ploiting **Perio**dicity for Opportunistic Energy-Efficient Data Transmission

Philipp Kindt, Han Jing, Nadja Peters and Samarjit Chakraborty
Institute for Real-Time Computer Systems (RCS), Technische Universität München TUM)
kindt/han/peters/chakraborty (at) rcs.ei.tum.de

***Abstract*—Reducing the energy consumption to the minimum is a crucial design requirement for all body area sensor networks. Sensors deployed on the human body, especially at the limbs often move along different positions. Usually, the transmit power is set to a sufficiently high value to achieve reliable transmission for the constellation with highest attenuation. For periodic movements, data transmission can be carried out at the position of the lowest path loss between the sender and the receiver, provided this position can be reliably identified. We propose a novel framework that predicts this position using acceleration data and the received signal strength. By learning a correlation between these signals, accurate predictions can be performed and up to 24.7% of the power spent by a Bluetooth Low Energy module for the transmission of a packet can be saved while still achieving the same packet error rate as with sending using the higher transmit power.**

I. Introduction

One of the main constraints of body area sensor networks (BASN) is battery run-time, imposing the need to reduce the power consumption of the nodes. Several approaches to reduce the power consumed have been presented in this active field of research. These are based on energy efficient routing schemes, maximizing the time a node is in sleep mode, reducing the amount of data transferred between the nodes and optimizing the topology of the network [1]. By optimizing the topology of a BASN, the nodes can reduce their transmit power level to reach the rest of the nodes. For these strategies, nodes that are moving quickly are a challenge, as the required transmit power is hard to predict. The received signal strengths of motes that are deployed on moving body parts, such as the arms or the legs, vary strongly over time, as the distances between the motes increase and shrink repeatedly. In addition, sometimes the line of sight between the sender and the receiver is interrupted by parts of the body, causing a high attenuation of the RF signals.

In this paper, we envision a scenario where multiple small sensors are worn on a human body. These sensors could be integrated into the clothing, the shoes or into other accessories. The number, types and configurations of these sensors might change depending on what the person is wearing. The system needs to adjust itself seamlessly whenever the setup is modified. These sensors send their data to a smartphone being carried by the user. For example, the phone could analyze accelerometer signals from the sensors to derive gait patterns, step frequencies or other fitness parameters. In such a scenario, the communication between the sensors and the phone is inevitable and the signal processing needs to be done on the phone since it is the only device to which the signals

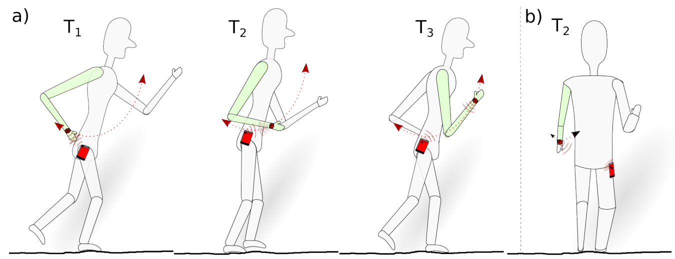

Fig. 1. Body-area sensor network consisting of an acceleration sensor on the wrist and a smartphone in the trouser pocket at three different points in time with a) phone and sensor on the same body side and b) opposing sides. In Figure 1b), the body interrupts the line of sight between the devices.

from all sensors are available. In addition, in spite of sensors having increasing computational capabilities, their resources are still not sufficient for many applications. Also, battery lifes of sensors are important, imposing the need to offload as many computations as possible onto the smartphone. Also devices which process their data locally often need to send the processed results to a smartphone in order to be displayed or stored. The purpose of our proposed scheme is to save transmit power of the sensors sending their data to the smartphone.

As already mentioned, the data of these sensors is sent via a wireless link to a smartphone worn in the trouser pocket, e.g. to perform a precise analysis of the walking pattern or to monitor the physical fitness. Obviously, the distance between the sensors and the phone increases and shrinks periodically for many on-body locations such as the arms and the legs. For higher distances, increased transmit powers are required. In addition, as depicted in Figure 1b), the line-of-sight between a sender and the corresponding receiver can be interrupted by parts of the body in some positions. This happens for example if the smart phone is in the pocket opposite to an arm a sensor is worn on. In these cases, the human body in between the devices causes a high attenuation of the signals transmitted, imposing the need for high transmit powers to communicate reliably. Typically, this power is chosen such that it is strong enough to achieve reliable transmission for the largest distance that might occur within the setup. For all other positions, energy is wasted. However, if the points with low attenuation could be *predicted*, the senders and the receiver could wakeup only at these points in time and perform the communication. With such a scheme, a significant amount of energy could be saved. As the movements of the human body are not perfectly harmonic, predicting these points is challenging. In this paper, we propose a novel scheme to exploit the periodicity of movements. Rather than sending the data from arbitrary locations, we attempt to predict the points with lowest path loss using acceleration data and the learned received signal strength indicator (RSSI) values of the past. By establishing a correlation between these signals, acceleration data can be used to make robust estimations of the predicted closest points. If no exploitable periodicity is present, the system can exchange packets in the usual fashion as without ExPerio. Based on the predictions, data is sent using a reduced transmit power. Our experiments show that significant energy savings can be achieved while keeping the error rate low. For a Bluetooth Low Energy (BLE) module, up to 24.7% of the power needed for the transmission of a packet can be saved with our scheme. For the sake of simplicity of exposition, we consider the scenario depicted in Figure 1 in the following. It consists of a person wearing a sensor at the wrist (e.g., a fitness watch) and the smartphone in one of its trouser pockets. We will use this as a running example throughout the paper. However, the techniques we present are general and can be applied to other setups as well. Whereas the scenario described above is analyzed in detail in this paper, there are many other applications that can benefit from the proposed scheme. For example, as described in [2], periodic signal attenuation occurs between many on-body-locations. Hence, most BASN are expected to derive a benefit from our predictive technique. In addition, also body-to-infrastructure scenarios can take advantage of ExPerio. Consider a wireless sensor wristband for swimming. At frequencies of $2.4GHz$ used by BLE and similar protocols, water has a high attenuation of $257dB/m$ [3]. The sensor can send its data to a smartphone outside of the water only above the surface. When a swimmer is doing the crawl, the device would have to repeat a packet until it is out of the water. With our technique, these points could be predicted and hence high amounts of energy could be saved.

Our Contributions: Compared to the related work, our main contributions are:

- We propose ExPerio, a novel scheme to **ex**ploit **perio**dic movements to reduce the energy consumption of wireless links in BASN by predicting points in time for transmitting data with low path loss.
- We evaluate our proposed scheme with experiments for two case studies and perform a feasibility analysis and an estimation of the possible power-savings using this technique.

The rest of this paper is organized as follows. Related work is presented in Section II. In Section III, we present our proposed scheme to exploit periodic motion in BASN. In Section IV, we verify the performance of ExPerio based on a BLE sensor and estimate the power savings that can be achieved by its application. Afterwards, the computational complexity is evaluated. Finally, we conclude our work and give an outlook on further research that needs to be done.

II. Related Work

The farther a signal is transmitted, the more power needs to be spent by its sender. Recently, several attempts have been made to exploit this correlation in intelligent power adjustment schemes, in order to minimize the signal transmit power. In [4], Correia et al. introduced two medium access (MAC) protocols for the Mica Motes 2 platform. The first one adjusts the transmit power by iteratively sending query packets through the 22 available power levels of the motes. In the second method, the receiving node calculates the optimal transmit power based on the RSSI and the noise of the surrounding. In [5], energy efficient topologies for scenarios, where multiple nodes send the same packet simultaneously to the receiver have been studied. By optimizing these topologies, energy can be saved. However, these approaches do not take periodic movements into account.

As our proposed scheme exploits periodicity, it relies on detecting suitable phases of periodic motion and extracting the fundamental frequencies of the recurrent movements. To realize frequency extraction, many solutions have been proposed in the past. For example, in [6], Gams et al. presented a technique to learn the phase and the frequency of a periodic movement online. They used a set of adaptive phase oscillators with a feedback structure to extract the properties of a signal. Moreover, wavelet based analysis is commonly used in the field of gait recognition [7].

Towards low power body-area sensor networks, a high signal attenuation within the human body for electromagnetic waves especially in the 2.4 GHz band has been studied extensively in recent literature. In [8], a propagation model for radiation in different frequency bands in human tissue of the gastro-intestine tract was presented. The results show a path loss of approx. $14dB$ in 0.1 meters of tissue. In [9], the path loss between 2.45 GHz-transceivers located on different body parts was measured. These measurements revealed a path loss of $19.2dB$ for the trunk of a body. The strong in-body attenuation requires a high power to be spent on transmission. Hence, techniques to overcome the high attenuation of the human body are needed to save power.

As expected, periodicity in the signal strength has also been observed before (e.g., in [2]). Previous works have assumed periodicities and proposed wake-up receiver-based approaches to exploit them [2], or attempted to make predictions based on the signal strength, only [10]. An approach which was developed in parallel with ExPerio has been presented in [11]. Like with ExPerio, an accelerometer has been used for predicting beneficial points in time for opportunistic on-body communication. Unlike ExPerio, which has been evaluated with BLE, IEEE 802.15.4 has been used. ExPerio relies mainly on peak detection, whereas [11] proposes dynamic time warping (DTW) and autocorrelation-based methods. Networks with multiple nodes on different body parts have been considered and a congestion-aware packet scheduling scheme has been proposed. In [11], the system was evaluated considering the improvement of the packet reception rate (PRR) when the transmit power was fixed, the queueing delay and the energy overhead for processing and signaling. However, no study on the amount of possible transmit power reduction for a given maximum PRR has been performed. In [11], it is concluded that such an investigation (which is presented in this paper) would be beneficial.

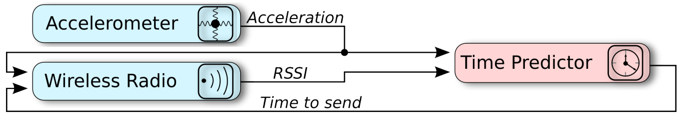

Fig. 2. Building blocks of ExPerio

III. ExPerio

A. System Overview

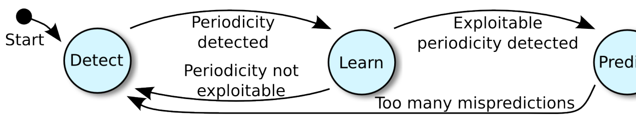

Fig. 3. State diagram of the ExPerio predictor

As mentioned earlier, in order to illustrate our proposed technique, we will use a setup with a wrist-worn acceleration sensor and a smartphone as a running example.

In this setup, a system as depicted in Figure 2 is implemented on the sensor. Acceleration data from a three-axis accelerometer (or any other payload) is sent via a wireless radio. A predictor module analyzes it together with the RSSI to predict the best point in time to send out the data. The RSSI signal is the received power in dBm. Our proposed system uses a BLE112-transceiver [12]. BLE is a time-slotted master-slave protocol. The data transmission is organized in so-called connection events that occur periodically. In each event, pairs of packets from both sides can be exchanged. For further details on BLE, we refer to [13]. The sensor is a BLE slave, whereas the smartphone acts as a master. However, the scheme can be adopted to a broad range of wireless technologies and protocols. The task of the predictor is to predict the time instants with lowest path loss in the wireless link, based on learned data from the recent past. At these predicted times, packets can be sent with reduced power compared to the power needed for sending at random points in time. As shown in Figure 3, the predictor consists of a state machine with three states. In each state, different algorithms are executed. In the rest of this section, we describe the signal processing algorithms for each state.

B. Prediction Algorithm

1) Detection state: The detection state is depicted in the first row of Figure 4. In this state, data transmission is carried out in the usual fashion, as it would have been done without ExPerio. At the same time, the slave tries to detect if there is periodic motion that can be exploited. For this purpose, the acceleration signals of three axes, a_x, a_y and a_z are analyzed. First, a_x, a_y and a_z are filtered by a second-order low-pass butterworth-filter (LPF) which has a cutoff frequency of $5Hz$. The sensor noise is removed and the signals are smoothened for extracting the fundamental frequency f_0, which is considered the most important frequency component representing the waveform. In our evaluation implementation, FFT is used as a baseline technique for our study since it is robust and easy to implement. However, other methods for fundamental frequency extraction which have lower computational complexity can be used as well. The fundamental frequencies $f_{0,x}$, $f_{0,y}$ and $f_{0,z}$ are determined by identifying the Fourier coefficients having the highest value for each axis. The filtered signals are then examined for periodicities. Signals with different fundamental frequencies on all three axes are considered as aperiodic. If the fundamental frequency coincides on at least two axes, it is checked by the periodicity detection block whether it is in a suitable range for periodic movements of the human body. If the condition $0.25Hz \leq f_0 \leq 5Hz$ is fulfilled, the motion is considered as periodic and the axis selection is activated. To select the axis which is best suited, the minimum and maximum amplitudes of the accelerations on all axes are detected. The axis selection block selects the axis with the highest signal amplitude by subtracting the minimum from the maximum acceleration. The selected axis is denoted as axis S with its corresponding fundamental frequency f_0. From experiments we noticed that in some cases f_0 on one axis is $2\times$ the frequency on another axis because of the different directions during periodic motion. This is equivalent with two acceleration peaks existing in one motion period. For all further steps of the algorithm, we remember whether the selected axis has a fundamental frequency f_0 which is $2\times$ that of another periodic axis or not.

If the conditions described above are fulfilled, periodic motion is assumed to be present. To explore if it can be exploited to save power, the learning state is initiated. The steps described above are performed on a set of samples containing the data of the last 10 seconds. If no periodic motion could be detected, another examination is performed after 1 second. If there is no periodic motion present, precautions need to be taken in order to avoid the detection phase being executed repeatedly. A possible solution could be pre-filtering of the accelerometer signal (e.g, disactivation of ExPerio in the case of small acceleration amplitudes).

2) Learning state: In the learning state, packets are received with small duty-cycles in between to allow for quasi-continuous sampling of RSSI values. For example, 17 packets

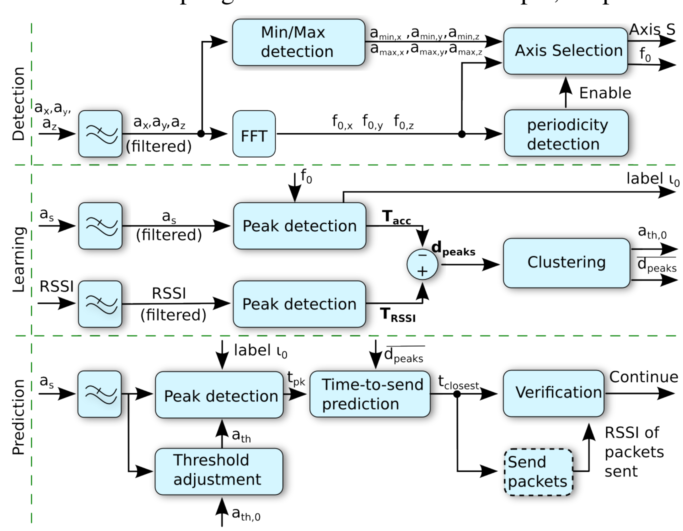

Fig. 4. Signal processing blocks in the 3 states of ExPerio

per second are received from the master to correlate the RSSI signals received along with the acceleration values a_s of the selected axis S. From this data, the algorithm detects whether the periodic motion can be exploited to save power. In addition, statistical data for the prediction ($\overline{d_{peaks}}$) is extracted. The steps involved in this process are executed on a set of samples that contain the data of 10 seconds. They are depicted in Figure 4 and are described below.

After the LPF, the peaks of both a_s and the RSSI signal are detected. The peak detection of a_s is based on its fundamental frequency f_0 using a similar method as described in [14]. At first, a period of $1.2 \times \frac{1}{f_0}$ in the middle of the set of samples is examined. This time-interval is chosen to be slightly longer than one signal-period to ensure that exactly one peak exists within it. The maximum acceleration value it contains is considered as the first peak. The samples that are neighboring this peak are discarded and the next peaks are searched for in a small area around $T_0 = \frac{1}{f_0}$ before and after this peak using the same method. The length of this area is empirically chosen to be $0.5 \times T_0$. This process is repeated until all the peaks have been found. Consequently, only half of the samples need to be examined and hence the computational complexity is reduced. Peaks of the RSSI signals are detected accordingly.

As already mentioned, there are cases where two acceleration peaks exist within one gait cycle. Such cases are handled as follows. The first accelerationpeak that is detected in the learning state is chosen as a peak of reference. This peak is labeled as a reference-peak (*r*-peak). Since the next peak corresponds to the same gait cycle, it is labeled as a non-reference-peak (*n*-peak). This labeling-scheme is repeated for all further peaks. Every *r*-peak is related to its adjacent RSSI peak and all r-peaks ideally have the same temporal distance to it, whereas *n*-peaks have different common distances to their corresponding RSSI peaks. Therefore, only *r*-peaks are considered for further processing.

After the peak detection, two arrays $\mathbf{T_{acc}}$ and $\mathbf{T_{RSSI}}$ containing the points in time of the detected acceleration- and RSSI-peaks are obtained. Then $\mathbf{d_{peaks}}$, the series of differences of each pair from $\mathbf{T_{RSSI}}$ and $\mathbf{T_{acc}}$, is calculated by $\mathbf{d_{peaks}} = \mathbf{T_{RSSI}} - \mathbf{T_{acc}}$. The label ι_0 of the last peak detected in the learning state is used as the starting label for the prediction state. The values of $\mathbf{d_{peaks}}$ are clustered with a hierarchical clustering method [15]. Each value $\mathbf{d_{peaks}}[n]$ starts in its own cluster and pairs of clusters are merged such that the distance between two clusters is minimized. Clustering stops when the distances between the clusters exceed a certain threshold. From the cluster containing most of the samples of $\mathbf{d_{peaks}}$, the mean value $\overline{d_{peaks}}$ of all contained samples is computed. This parameter establishes a relationship between a_s and the RSSI signal. In particular, it contains the average time between maximum acceleration and best reception. Values from other clusters are considered as irregular and are discarded. Before changing to the prediction state, an initial acceleration threshold $a_{th,0}$ used in the prediction state is computed adaptively based on the peaks of a_s using $a_{th,0} = \delta(\widetilde{a_{T_{acc}}} - \overline{a_s}) + \overline{a_s}$. In this equation, $\overline{a_s}$ is the average acceleration level of all samples examined and δ is a constant value, which we set to 0.5 in our implementation. $\widetilde{a_{T_{acc}}}$ is the median of the detected acceleration peak values. In some cases, the values of $\mathbf{d_{peaks}}$ in the selected cluster differ significantly and hence no correlation between the acceleration and the RSSI signal can be established. If the difference between the minimum and maximum value in the selected cluster exceeds a certain threshold, the learning process is repeated $1s$ later. The smaller this threshold is, the more precise the mean peak distance $\overline{d_{peaks}}$ becomes. However, a small threshold can result in longer learning phases. If the learning state could not succeed 10 times in a row, then the system returns to the detection state.

3) Prediction state: In the prediction state, a second order low pass filter with a cutoff frequency of 10 Hz smoothes the acceleration signal. The time for sending the next packet to the smartphone, $t_{closest}$, is determined as depicted in the last row of Figure 4. Unlike the previous states, the prediction state is not run on a fixed set of samples, but operates in real-time on a per-sample basis, as quick decisions on when to send a packet need to be taken. Hence, the steps described below are executed once for each acceleration sample on the selected axis. Because of the real-time requirements, peaks of a_s cannot be detected by the method described for the learning state, as the location of a peak needs to be determined before the whole period has passed. Hence, the peak must be detected as soon as it occurs. A sample $a_s[n]$ is considered to be a peak if the following inequalities are fulfilled:

$$a_s[n] > max\{a_s[n-n_p], a_s[n+n_p]\} \cap a_s[n] > a_{th} \quad (1)$$

The term n_p makes the peak detection more robust against noise. However, the peak can only be detected n_p samples after it has occurred. In cases where the point of best reception is very close to the acceleration maximum, high values of n_p would lead to a time offset between the predicted and the real optimal time to send the packet. Therefore, n_p should be kept low. In our implementation, we used a value of $n_p = 5$. After a peak, some samples are skipped and the next sample is examined for further peaks after $\alpha \times T_0$ time units. α is a constant value which we empirically chose to be 0.7 in our implementation. For a_{th}, an adaptive threshold adjustment is performed. Its initial value is $a_{th,0}$ from the learning state. To adjust this threshold to the current situation, the mean value of all samples of the preceding three periods $\overline{a_{3p}}$ is computed. In addition, the median acceleration of the three last recent peaks $\widetilde{a_{m3p}}$ is computed. With this data, the threshold is calculated using a percentage p_{th} as in the equation below.

$$a_{th} = \overline{a_{3p}} + p_{th}(\widetilde{a_{m3p}} - \overline{a_{3p}})$$

For computing $\widetilde{a_{m3p}}$, rather than reusing peaks previously detected with Equation 1, they are re-detected with a method similar to the one described for the learning state. Small sets of samples around the previously detected peaks a_{pk} are examined and for each of these sets, the point in time with the maximum acceleration is considered to be the peak. This provides a more robust detection than those computed in real-time. In addition, the detected peaks are labeled with *r* and *n* as in the learning state. In case a peak could not be detected, the labels are corrected backwards to avoid mispredictions based on the time of the next detected peak. For example, if a peak is labeled with *n*, but the time measured from the previous detected peak is approximately $\frac{1}{f_0}$, its label is corrected to *r*, as obviously a peak has been lost in the meantime. For the case that the periodic motion stops, we define a truncation criterion as follows. If no acceleration peak is detected for slightly longer than one period (e.g $1.2 \times T_0$), then the threshold a_{th} is decreased by a constant value (e.g. $a_{th} = a_{th} - 0.04g$) to allow the system to recover. The interval is chosen slightly longer than T_0 to account for cases when the motion period in the prediction state is increased compared to the learning state. If no acceleration peak is detected for more than $2 \times N_{mp,max} \times T_0$ time units, at least two periods have been lost and the system goes back to the detection state.

Using the time of the last detected peak t_{pk}, the time with best reception ("closest point") is predicted as $t_{closest} = t_{pk} + \overline{d_{peaks}}$. At $t_{closest}$, a packet is sent from the sensor to the phone and the phone sends the received RSSI back to the device. For verification, this RSSI is compared to a threshold value to verify the prediction accuracy. If the RSSI is below it, the prediction is considered to be a misprediction. If this has happened more than $N_{mp,max} = 3$ times in a row, the prediction state is left and another detection phase begins.

IV. Validation

In this section, we demonstrate and verify our system in our previously described example scenario, with BLE as the communication protocol. The goal is to describe a concrete application scenario that uses the proposed algorithm and to evaluate the robustness of the prediction and the resulting power savings. To accomplish this, we perform a combination of simulation, analysis and measurements.

A. Setup

In our setup, a wrist-worn sensor sends 100 accelerometer values per second for each of its 3 axes to a smartphone in the pocket. The accelerometer used is a LIS302DL [16] MEMS accelerometer that creates samples with a rate of $100Hz$. In the scenario described, we send all samples to the smartphone. However, our proposed scheme is not limited to this usecase. Usecases such as mobile health monitoring (e.g, oxygen saturation, pulse rate) sometimes require raw data to be sent to the smart phone. In addition, devices that process data locally (e.g, step counters) need to send their data to be displayed and stored as well. Different applications result in different numbers of packets per event, packet-sizes and packet exchange frequencies. Our scheme can be adapted for these situations as well, however, the resulting energy savings might vary.

The sensor used for our experiments consists of a PCB containing a microcontroller (ARM Cortex M), a 3-axis-accelerometer and a BLE112 Bluetooth LE module. This demo-system can be attached to the wrist. For emulating the smartphone, we place a BLED112 USB dongle in the trouser pocket, which is connected to a laptop using a sufficiently long wire. This allows a test person to move freely, while a second person follows with the laptop. The dongle in the pocket acts as the BLE master, whereas the test board at the wrist as a slave. On the laptop, a software we developed receives the acceleration values, measures the RSSI of the signal received and writes it onto the laptops harddisk. The system is powered by a battery in the back pocket of the person wearing it.

In our scenario, the antenna of the sensor is on the side of the arm that is facing the body by attaching the board to the inner side of the wrist. This is equivalent to an antenna (e.g. of a fitness watch) being in a wristband. Consequently, the arm is not in between sender and receiver and less power is needed when both are within a line of sight. We then studied two cases:

- **Case 1)** The smartphone is in the trouser pocket at the same side of the arm the sensor is worn on.
- **Case 2)** The smartphone is in the pocket at the opposite side of the arm the sensor is worn on, resulting in a high signal attenuation during transmission through the body.

On the MCU of the test board, the following procedure is run:

1) Record 6 acceleration samples $a_x[n], a_y[n], a_z[n]$ for each axis (1 byte per sample). The accelerometer generates triples of values with $100Hz$.
2) Prepare the payload for a BLE packet consisting of the 18 bytes forming the 6 acceleration samples.
3) Add one byte consisting of the RSSI of the last packet received from the remote side.
4) Send the 19 bytes of payload prepared as a BLE packet (which can contain up to 20 bytes of payload).

When executed, these steps result in a series of BLE packets. The master sends a polling packet to the slave with a period of $T_c = 7.5ms$, whereas the slave has a packet with 3×6 acceleration samples ready to send each $(\frac{100Hz}{6})^{-1}$ seconds. The slave sends it as a response to a polling packet. On a laptop, this data stream is recorded and imported into MATLAB for further analysis.

B. Received Signal Strength Requirements

To evaluate the performance of ExPerio, first, the minimum received signal power needed by the smartphone for successful reception was determined. The wider bars in Figure 5 show the error rate for approximately $180,000$ received packets with different RSSI values. The sensor was attached to the wrist and sent advertising packets. These packets were received by the BLE dongle in the pocket that executed a packet sniffer firmware [17] to receive packets regardless of cyclic redundancy check (CRC) mismatches. The sensor was moved slowly along the path according to Case 2) as described above. In addition, the experiment was repeated with different transmit power levels. The packet sniffer recorded the received power and computed the number of CRC errors for each RSSI value.

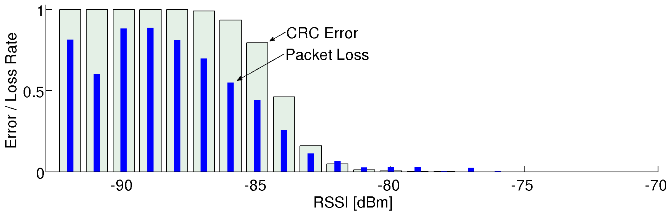

Fig. 5. Packet error and loss rate over RSSI for approx. 180,000 packets

As can be seen in Figure 5, for received signal strengths above $-80dBm$, almost no packet errors occur. Below $-80dBm$, the error rate quickly increases to almost 100% at $-88dBm$. The experiment has been carried out using BLE advertising packets[1] whereas accelerometer values from the test setup were sent using BLE data packets. Therefore, between the packets shown in Figure 5 and the data packets of ExPerio, there are three main differences: 1) Instead of one channel, packets are sent on different channels following a hopping sequence [13]. The receiver sensitivity varies by about $0.4dBm$ for different frequencies [18] resulting in slightly varying error rates for different packets. 2) The packet lengths might be different. As the bit-error-rate remains constant, packets consisting of more bits have a higher packet error rate. In our experiment, the advertising packets had a length of 37 bytes, which is the same length of the payload packets in the scenario described. The CRC does not cover all bytes of the packet [13]. 3) Packets are sent in response to a polling packet. If a polling packet is lost, its response will be lost as well. As the influence of these differences is limited, a simple error model for data packets can be derived from this curve.

We consider a packet as lost, if its RSSI is below or equal to $-80dBm$. Otherwise, we consider a packet as received successfully. This very conservative error model is used within the rest of this paper to evaluate our proposed scheme. A packet error is detected by a CRC mismatch. In addition to CRC errors, packets can get lost. A lost packet is never received by the packet sniffer. While the error rate can be measured directly, the loss rate in Figure 5 is a rough estimation. It has been obtained by dividing the interval between two consecutively received packets by the average time between the transmission of two advertising packets of the sender. If a packet was lost, the RSSI of the last packet received before the lost packet has been assigned to it. Consistent with the packet error rate, Figure 5 shows that there is no loss for RSSI values greater than $-80dBm$, whereas the loss rate quickly increases below this value.

C. Prediction Algorithm

To evaluate the performance of the two cases described, we attached the setup to the bodies of 6 different subjects and recorded the streams received by the master for two minutes per experiment. During this time, the subjects walked along a nearly straight corridor, turning around at its ends. A MATLAB implementation of our algorithm used the recorded data to evaluate the performance of ExPerio. In each of the states, the algorithm only used the information that would have been available in an online-implementation running on the sensor, as described in Section IV-D. After 10 seconds of detection and another 10 seconds of learning phase, the prediction phase was started. In some cases, the detection or learning phase took slightly longer. Figure 6 shows the RSSI in the prediction phase for both of the cases. For Case 1), where sensor and phone were on the same side of the body, the packets were sent with a transmit power of $-6dBm$. For Case 2), the packets were sent with a power of $3dBm$, which is the maximum transmit power the device supports. The transmit power was chosen high enough to almost completely avoid the loss of packets or CRC errors in Case 1), or to limit the packet error rate to its lowest possible value in Case 2). The predicted points in time $t_{closest}$ when packets would have been sent by an online-implementation of ExPerio have been determined by the MATLAB implementation. We consider a prediction as correct, if the RSSI is above a predefined threshold of $-70dBm$. The value of this threshold determines how frequently ExPerio goes back to the detection state and should be chosen as a certain percentage of the maximum RSSI in online-implementations. In Figure 6, a right prediction is marked with a dot, while a misprediction is marked with a cross. As can be seen, for all subjects the prediction state has never been left because more than $N_{mp,max} = 3$ mispredictions in a row never occurred. For all subjects and for both cases (except for Subject 2 in Case 1)), the average received power per packet is strongly increased by the predictions of ExPerio. For Subject 2 in Case 1), the prediction did not work well because the subject moved less periodically in the prediction state than in the learning state. The threshold for mispredictions of $-70dBm$ is not well chosen for this experiment - the system should have detected the situation and should have gone back to the detection state until the movements have become sufficiently periodic again. Another observation when zooming into Figure 6 is that for Case 1), the RSSI peaks are very narrow, as the hand always has a high velocity when it is close to the sender. In Case 2), the peaks are much broader so predicting times within these peaks is easier. Even though making good predictions is harder for Case 1) than for Case 2), our proposed algorithm worked robust for Case 1), too. As can be seen in the figure, in some of the experiments the detection or learning phase took slightly longer than 10 seconds and hence the prediction state started later.

	Case 1 [dBm]			Case 2 [dBm]		
$p_{e,max}$	1%	3%	5%	1%	3%	5%
Subj. 1	26	23	21	3 [2]	9 [3]	14 [3]
Subj. 2	-4 [4]	-2 [4]	3	18	16	16
Subj. 3	11	18	18	28 [3]	25	23
Subj. 4	29 [3]	26	25	21 [3]	20 [3]	18 [3]
Subj. 5	7	12	9	2	1	0
Subj. 6	0	4	3	23 [3]	19 [3]	19

TABLE I
POSSIBLE REDUCTION OF THE TRANSMIT POWER WITH EXPERIO IN dBm FOR DIFFERENT MAXIMUM ERROR RATES

[1]*Advertising packets* in BLE are used for connectionless communication. We refer to packets sent for connected data transmission as *data-* or *payload packets*.

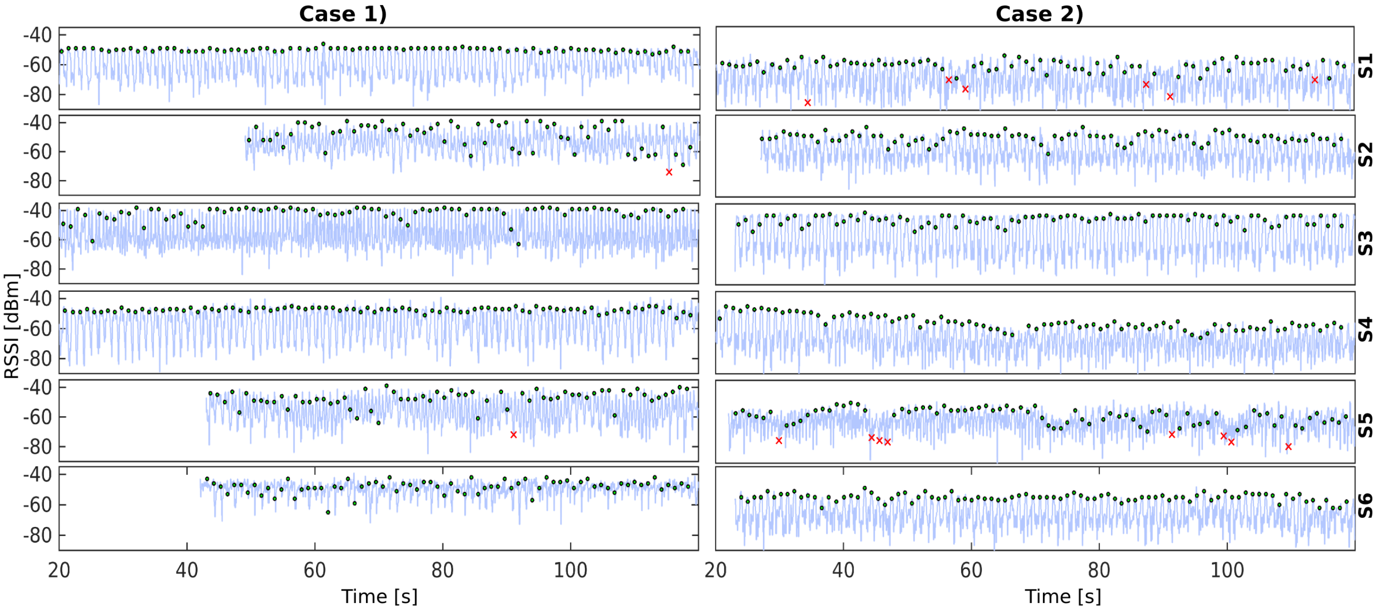

Fig. 6. Recorded RSSI over time for 6 different subjects for sender and receiver at same body side (Case 1) or on opposite sides (Case 2). The circles and crosses depict the predicted times to send and their corresponding RSSI values. A circle depicts a successful prediction, a cross a misprediction. In some experiments, no mispredictions occurred.

By applying the error model described above, we can estimate the possible savings of transmit power with ExPerio. We assume that we wish to send with a certain maximal overall error rate $p_{e,max}$. The overall error rate is defined depending on the total number of packets N below and above a $-80dBm$ RSSI-threshold:

$$p_e = \frac{N|_{RSSI \leq -80dBm}}{N|_{RSSI > -80dBm} + N|_{RSSI \leq -80dBm}} \quad (2)$$

The error rate of the predicted packets, p'_e, is defined similarly by only taking predicted packets into account. By assuming that the path loss remains constant, we can simulate a reduction of the transmit power as follows. We substract a given amount of transmit power from all RSSI values in the data set recorded, while leaving the $-80dBm$ RSSI-threshold unchanged. In a simulation, we repeatedly reduced the transmit power and computed the resulting error rate with and without prediction. The results presented in the following are based on this simulation. If, according to the model, an error bound could not be reached with the transmit power the device used in the experiment, we simulated a transmit power increase by adding corresponding amounts of power to reach the error bound. For example, to achieve a maximum error rate of 5% for Subject 1, without ExPerio, the transmit power for Case 1) could be reduced from $-6dBm$ to $-12dBm$. With ExPerio, it could be further reduced by $21dBm$ from $-12dBm$ to $-33dBm$ while still achieving an error rate below 5%. Table I summarizes these savings for different maximum packet error rates $p_{e,max}$ for all the subjects in both cases. As can be seen, except for Subject 2 in Case 1), the transmit power could be reduced in all experiments. The results show that $t_{closest}$ can be predicted with high success rates from the acceleration signal. It needs to be pointed out that our experiments constituted a hard test case for ExPerio as the periodicity of the walking was disturbed by the turnarounds at the ends of the corridor. Nevertheless, the prediction worked reliably. Further experiments we made showed similar results. The table shows that higher tolerated packet error rates $p_{e,max}$ do not always lead to higher transmit power reductions. Higher values of $p_{e,max}$ lead to higher possible reductions using ExPerio, but lead to higher possible transmit power reductions when sending without ExPerio, too. Another interesting outcome is that in some of the experiments, up to two r-labeled peaks have been missed (and hence sending opportunities have been missed). However, these misses have been detected and corrected by the algorithm such that the prediction phase has not been aborted.

[2]This error rate could not be achieved neither with nor without ExPerio as the transmit power in the experiment was too low. In the simulation, the tx-power was increased to satisfy the 1%-bound with and without ExPerio.

[3]Without ExPerio, a higher transmit power was needed to achieve the given error rate. With ExPerio, the bound could be satisfied.

[4]Due to many mispredictions, a higher transmit power was needed with ExPerio than without.

D. Power savings

In this section, we estimate the possible savings in a device's current consumption that can be achieved by the application of ExPerio. If a constant voltage is supplied to the BLE module, the power- and energy-consumption of the chip is proportional to its current consumption. The current savings presented in this section are computed using the estimation technique described below that combines measured data with simulation results. For assessing the energy saved for transmission using ExPerio, we first describe how an online-implementation of our algorithm would make use of the BLE protocol in an energy-efficient manner. As shown in [19], the most energy-efficient way to send data with BLE is to send as many packets as possible in one connection event. Hence, one would send the acceleration values in bursts that occur with a given period to spend as little power as possible. Under the assumption that 5 packets can be sent within one connection event (which holds true for the BLE112 module), given a motion period T_0 of around 1.17 seconds [20], 4 connection events with 20 payload bytes would be needed to send $3 \cdot 117$ acceleration samples per period. The first 3 events would contain 100 bytes of payload, the fourth the remaining 51 bytes. With ExPerio, the system would act as follows. If no periodicity was detected, we would send these 4 bursts of packets at arbitrary points in time with a transmit power high enough so that the packets would be received from any position of the sensor. If periodicity was detected, the transmit power would be lowered to the level needed for the position with best reception and the packets would be sent only at the predicted times. 4 consecutive connection events that occur shortly after the predicted time would have been used to send out the packets. We assume that the RSSI does not change significantly within sending these 4 consecutive connection events and therefore has the value of the first event for all 4 ones.

The code running on our evaluation system generates a continuous stream of connection events sent with high power, rather than four discrete burst events. From this stream, we can reconstruct a series of packets that is similar to the optimal one that would be created by an online-implementation on the MCU. In particular, the signals are generated as described below: In the **detection state**, the RSSI signal is ignored and only a_x, a_y and a_z are evaluated. As the processing would be done on the sensor anyway, there is no difference between the emulation and an online implementation. In the **learning state**, the RSSI and acceleration signals are used. Here, the packet stream generated by the MCU is similar to the packet stream of an online-implementation. In the **prediction state**, we use the acceleration signals received. From the RSSI signal, only the value following the predicted time $t_{closest}$ is used to verify the prediction. Thereby, there are two differences from an online-implementation: First, in an online-implementation, connection events containing bursts of packets would be received by the phone. Each of these packets would have its own RSSI. One would calculate the average value for each event and then evaluate the results for all four events of one period. In the stream generated by the evaluation system, only one RSSI value for the packet sent shortly after $t_{closest}$ is available. However, as both events would occur with maximum 3 connection intervals ($22.5ms$) in between, the impact on the evaluation results is small. The second difference is the transmit power. In an online-implementation, as already described, the packets would be sent with reduced power. We simulate these reductions based on the recorded data, as described below.

1) Power considerations using BLE: To determine the energy savings, we calculate the module's current consumption using a BLE energy model based on [21] and [19], which has been extended by further measurements. For the simplicity of explanation, we describe how ExPerio reduces the energy consumption of the scenario described by using Subject 3 as an example. Afterwards, we present the current savings for all subjects.

	Case 1			Case 2		
$\mathbf{p_{e,max}}$	**1%**	**3%**	**5%**	**1%**	**3%**	**5%**
Subj. 1	3.9%	3.0%	2.5%	0.0%	6.2%	11.0%
Subj. 2	−0.9%	0.0%	0.0% 2	12.6%	4.7%	4.1%
Subj. 3	1.2%	2.5%	2.1%	12.9%	10.7%	7.8%
Subj. 4	5.9%	3.3%	3.0%	11.8%	12.4%	12.4%
Subj. 5	1.5%	2.5%	2.2%	0.0%	2.0%	0.0%
Subj. 6	0.0%	0.0%	0.0%	12.4%	12.4%	12.6%

TABLE II
POSSIBLE REDUCTION OF THE BLE MODULE'S CURRENT WITH EXPERIO FOR DIFFERENT MAXIMUM ERROR RATES

In the current draw of a BLE module, there is a distinct peak caused by the transmission of a packet. Reducing the transmit power results in a lower amplitude of this peak. Its maximum possible reduction (changing the transmit power from $3dBm$ to $-23dBm$) is 27.8%. For example, for Subject 3 in Case 2) with a maximum packet error rate of 1%, this peak could be reduced by 24.7%. However, when considering the average current, other parts of the waveform remain constant and limit the overall current consumption. Using the BLE energy model, assuming that all data packets bear an overhead of 17 bytes and all response packets have 10 bytes length, the average current consumptions for the two cases can be computed[5]. The computations are performed by aligning the transmit power reductions from Table I to the next highest level supported by the BLE112 module. For Subject 3 in Case 1) with a maximum error rate of 1%, 1.2% of the overall current can be saved. For an error rate of 3%, the savings are 2.5% and for $p_{e,max} = 5\%$, the savings are 2.1%. Higher savings can be achieved for Case 2). The values for Case 2) are 12.9%, 10.7% and 7.8%. The savings for all subjects are shown in Table II. For Case 2), the current consumption could be reduced for all subjects (assuming $p_{e,max} = 3\%$). For Case 1), where making good predictions is harder due to narrow peaks, the current consumption could be reduced for most subjects. In some of the experiments (e.g., Subject 1, Case 2)), the current could not be reduced even though the transmit power was lowered. The reason for this is the lack of the appropriate transmit power levels supported by the BLE module.

2) Power considerations using different protocols: Even though considerable savings can be achieved with BLE, the protocol is not suitable for exploiting periodicity perfectly. BLE requires that for each packet sent, one packet has to be received, even if there is no payload available from the remote device. As the amplitude of the reception peak is not reduced for different transmit powers, these peaks limit the achievable savings. Designing protocols that eliminate this requirement would allow higher savings. By discarding all parts of the current curve related to reception, the reduction for Subject 3 in Case 2) would be increased to approximately 19.6% with $p_{e,max} = 1\%$. The savings for such a protocol are shown in Table III. Compared to BLE, the savings are increased. Protocols that support higher numbers of packets

[5]Neglecting the packets that send the RSSI received by the phone back to the sensor which are slightly longer.

	Case 1			**Case 2**		
$\mathbf{p_{e,max}}$	$\mathbf{1\%}$	$\mathbf{3\%}$	$\mathbf{5\%}$	$\mathbf{1\%}$	$\mathbf{3\%}$	$\mathbf{5\%}$
Subj. 1	6.8%	5.0%	4.0%	0.0%	9.4%	16.3%
Subj. 2	−1.0%	−0.5%	0.5%	18.8%	7.5%	6.2%
Subj. 3	2.0%	4.0%	4.0%	19.6%	16.6%	12.4%
Subj. 4	9.5%	5.4%	5.0%	18.0%	18.8%	18.8%
Subj. 5	2.9%	4.0%	4.0%	0.0%	3.1%	0.0%
Subj. 6	0.0%	0.0%	0.0%	18.8%	18.8%	18.8%

TABLE III
POSSIBLE REDUCTION OF THE CURRENT OF A WIRELESS MODULE WITH A MODIFIED BLE PROTOCOL FOR DIFFERENT MAXIMUM ERROR RATES

per connection event, longer packet lengths, lower transmit power levels and optimized transmission schedules would further increase these savings considerably. In case of BLE, the standard [13] allows higher number of packets per event than the module we used supports - this would increase the current savings even for the BLE protocol.

E. Computational Cost

The savings we consider in this paper are related to the Bluetooth LE module only. For the predictions, there is an additional overhead caused by the processing of our algorithm on a CPU. The overhead clearly depends on the hardware platform and the sleep modes being inherent to it. Hence, we briefly discuss this by giving estimates on the number of operations on the sensor node. In the prediction state, the algorithm takes about 3000 arithmetic operations per second which is negligible. A learning phase that examines the samples of 10 seconds wall clock time takes about 36000 operations plus the overhead for clustering about 10 values - which can also be neglected. The detection state is dominated by the FFT we used in our evaluation implementation. Alternative ways for fundamental frequency extractions such as the ones presented in Section II should be used on embedded devices. Apart from the FFT, the overhead of the learning phase is < 100000 operations for the whole phase which analyzes a set of samples of 10 seconds.

V. CONCLUDING REMARKS

In this paper, we presented a scheme for exploiting periodic movements between sensors and receivers in BASN. As a plausibility analysis, we showed that considerable amounts of energy can be saved by its application. However, the possible savings are situation dependent. Advantageous situations can be detected automatically, so implementations of ExPerio could be activated only if there is a considerable gain in energy consumption. Alternative frequency extraction methods need to be evaluated and optimized communication protocols need to be developed to exploit periodicity more efficiently than with BLE. In addition, these protocols should also support current savings on the master side. This could be realized by the master making similar predictions of $t_{closest}$ to the ones made by the slave. Even though our results showed that ExPerio makes robust predictions, we believe that with more fine-tuning, its performance in terms of prediction reliability and computational overhead can be further improved.

The authors would like to thank the anonymous reviewers for their helpful comments. This work was partially supported by *HE2mT - High-Level Development Methods for Energy-Saving, Mobile Telemonitoring Systems*, a project funded by the federal ministry of education and research of Germany (BMBF) and the *EIT ICT labs* in the medical CPS activity, a project funded by the European Union.

REFERENCES

[1] S. Mahfoudh and P. Minet, “Survey of energy efficient strategies in wireless ad hoc and sensor networks,” in *Seventh International Conference on Networking (ICN)*, 2008.

[2] N. Roberts, S. Oh, and D. Wentzloff, “Exploiting channel periodicity in body sensor networks,” in *IEEE Journal on Emerging and Selected Topics In Circuits and Systems*, vol. 2, 2012, pp. 4–13.

[3] A. A. Abdou, A. Shaw, A. Mason, A. Al-Shamma’a, S. Wylie, and J. Cullen, “Wireless sensor network for underwater communication,” in *IET Conference on Wireless Sensor Systems (WSS)*, 2012.

[4] L. Correia, D. Macedo, and D. Silva, “Transmission power control in MAC protocols for wireless sensor networks,” in *8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems (MSWIM)*, 2005.

[5] Y. Zhu, M. Huang, S. Chen, and Y. Wang, “Cooperative energy spanners: Energy-efficient topology control in cooperative ad hoc networks,” in *IEEE International Conference on Computer Communications (INFOCOM)*, 2011.

[6] A. Gams, L. Righetti, A. Ijspeert, and J. Lenarčič, “A dynamical system for online learning of periodic movements of unknown waveform and frequency,” in *2nd Biennial IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2008.

[7] M. Sekine, T. Tamura, M. Akay, T. Fujimoto, and Y. Fukui, “Discrimination of walking patterns using wavelet-based fractal analysis,” *IEEE Transactions On Neural Systems And Rehabilitation Engineering*, 2002.

[8] L. Wang, L. Liu, C. Hu, and M.-H. Meng, “A novel rf-based propagation model with tissue absorption for location of the gi tract.” in *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2010.

[9] A. Alomainy, Y. Hao, A. Owadally, C. Parini, Y. Nechayev, C. Constantinou, and P. Hall, “Statistical analysis and performance evaluation for on-body radio propagation with microstrip patch antennas,” *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 1, pp. 245–248, 2007.

[10] K. Prabh and J. Hauer, “Opportunistic packet scheduling in body area networks,” in *European Conference on Wireless Sensor Networks (EWSN)*, 2011.

[11] J.-H. Hauer, “Leveraging human mobility for communication in body area networks,” *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, p. 39, 2014.

[12] *BLE112 data sheet*, Bluegiga, 2011, version 1.21, www.bluegiga.com.

[13] Bluetooth SIG, “Specification of the bluetooth system 4.0,” 2010, volume 0, www.bluetooth.org.

[14] M. Derawi, P. Bours, and K. Holien, “Improved cycle detection for accelerometer based gait authentication,” in *Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, 2010.

[15] B. Mirkin, *Clustering: A Data Recovery Approach, Second Edition*. Chapman and Hall/CRC, 2012.

[16] *LIS302DL MEMS motion sensor datasheet*, STMicroelectronics, 2008, revision 4, www.st.com.

[17] *SmartRF Packet Sniffer User Manual*, Texas Instruments, 2011, www.ti.com.

[18] *CC250 data sheet (2.4-GHz Bluetooth low energy System-on-Chip), revision F*, Texas Instruments, 2014, www.ti.com.

[19] M. Siekkinen, M. Hiienkari, J. Nurminen, and J. Nieminen, “How low energy is bluetooth low energy? comparative measurements with zigbee/802.15.4,” in *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2012.

[20] R. Tian, E. Du, K. Yang, P. Jiang, F. Jiang, Y. Chen, R. Sherony, and H. Takahashi, “Pilot study on pedestrian step frequency in naturalistic driving environment,” in *IEEE Intelligent Vehicles Symposium (IV)*, 2013.

[21] S. Kamath and J. Lindh, *Application Note AN092: Measuring Bluetooth Low Energy Power Consumption*, 2012, revision SWRA347a, http://www.ti.com/lit/an/swra347a/swra347a.pdf.