Department of Computer Science, University of Otago

Technical Report OUCS-2013-13

BWS: Beacon-driven Wake-up Scheme for Train Localization using Wireless Sensor Networks

Authors:

Adeel Javed, Haibo Zhang, and Zhiyi Huang
Department of Computer Science, University of Otago, New Zealand

Jeremiah Deng
Department of Information Science, University of Otago, New Zealand

---

Department of Computer Science,
University of Otago, PO Box 56, Dunedin, Otago, New Zealand

http://www.cs.otago.ac.nz/research/techreports.php

BWS: Beacon-driven Wake-up Scheme for Train Localization using Wireless Sensor Networks

Adeel Javed, Haibo Zhang, and Zhiyi Huang
Department of Computer Science
University of Otago, New Zealand
Email: {adeel, haibo, hzy}@cs.otago.ac.nz

Jeremiah Deng
Department of Information Science
University of Otago, New Zealand
Email: jeremiah.deng@otago.ac.nz

***Abstract*—Real-time train localization using wireless sensor networks (WSNs) offers huge benefits in terms of cost reduction and safety enhancement in railway environments. A challenging problem in WSN-based train localization is how to guarantee timely communication between the anchor sensors deployed along the track and the gateway deployed on the train with minimum energy consumption. This paper presents an energy-efficient scheme for timely communication between the gateway and the anchor sensors, in which each anchor sensor runs an asynchronous duty-cycling protocol to conserve energy and wakes up only when it goes into the communication range of the gateway on the train. A beacon-driven wake-up scheme is designed and we establish the upper bound on the amount of time that an anchor sensor can sleep in one duty cycle to guarantee timely wake up once a train approaches. We also give a thorough theoretical analysis for the energy efficiency of our scheme and give the optimal amount of time that an anchor sensor should sleep in terms of minimizing the total energy consumption at each anchor sensor. We evaluate the performance of our scheme through simulations. Simulation results show that our scheme can timely wake up anchors sensors at a very low cost on energy consumption.**

Keywords*—*wireless sensor networks, Localization, wake-up scheme.

I. Introduction

The world's railway infrastructure has been experiencing substantial growth in the last two decades. This growth imposes much greater pressure on railway operators due to the need to guarantee the safety of railway transportation. Real-time train localization plays an important role in achieving high-level railway safety and reliability, since it can assist the signalling system to initiate traffic signals, activate crossing gates, send alarms to trackside workers, and so on. Even though many GPS-based approaches have been designed and deployed for localization and tracking applications, drawbacks such as limited coverage and sophisticated infrastructures have prevented them from being used for train localization. For example, trains in subway systems primarily operate underground. Even in the above-ground railway transportation, trains may frequently pass through GPS dark territories such as tunnels and hilly regions [1]. Other technologies like WLAN, RFIDs and bluetooth also have certain limitation for train localization scenario due to either high cost of infrastructure installation and complicated protocol stack, or limited transmission range and lack of collision avoidance with limited computational capabilities.

Wireless sensor networking (WSN) is a promising technology for real-time train localization due to its huge benefits such as low cost, large coverage, and easy deployment/maintenance. Anchor sensors with hardcoded global coordinates can be deployed along the tracks to detect train movement. When the train is approaching, the anchor sensors can report their locations to the gateway installed on the train. Based on the locations of the anchor sensors as well as the Received Signal Strength (RSS) for transmissions from anchor sensors to the gateway, some localization schemes such as particle filter [2] can be used to estimate the train location in a real-time manner.

It is worth noting that several factors may influence the value of the RSS received by the gateway, e.g., presence of obstacles, interference, noise. Since particle filter uses RSS values and the location information sent by the anchor sensors, the more information provided by the anchor sensors, the more accuracy of the localization the particle filter will provide. However, the focus of this work is to ensure anchor sensors to timely communicate with the gateway sensor with maximum energy saving, which is crucial to the particle-filter-based train localization.

The accuracy of train localization heavily depends on the timely communication between the anchor sensors and the gateway on the train. At any time period, each anchor sensor that is within the communication range of the gateway stays in active state and reports to the gateway quickly as a train can move very fast. A straightforward approach is to let all sensors turn their radio transceivers on and work in idle listening state to detect the coming of the train. However, sensor devices are commonly powered by batteries, and this approach can quickly deplete the battery energy at each anchor sensor as idle listening consumes almost the same amount of energy as data transmission. Even though extensive work has been done on investigating energy harvesting techniques by using external energy sources such as solar or vibration energy, these techniques either demand large sized generators or can only produce power less than 100 μW [3], [4]. Providing power through the AC power lines is also not an efficient approach as it will be very expensive to deploy and maintain the infrastructure of the AC power lines along the tracks.

Duty-cycling is commonly used in WSNs to save energy by periodically turning radio off. A major challenge in applying duty-cycling for train localization is to guarantee that each anchor sensor can wake up in time once it goes into the communication range of the gateway. Even though many sensor wake-up schemes have been proposed, they are not suitable for real-time train localization due to the special features of railway environments. In this paper we focus on designing an efficient wake-up scheme which can guarantee that each

anchor sensor can wake up timely for communication with the gateway using the minimum energy. The main contributions of this work are summarized as follows:

- We proposed a beacon-driven anchor sensor wake-up model in which the gateway on the train continually broadcasts beacon packets to wake up anchor sensors that are going to communicate with the train. To guarantee that an anchor sensor can wake up in time, we first derive the upper bound on the amount of sleep time in one duty-cycle, and then design a beacon-driven anchor sensor wake-up protocol.
- We analyzed the energy efficiency of our scheme, and gave the optimal setting for the amount of sleep time in one duty-cycle in terms of minimizing total energy consumption at each anchor sensor node.
- We evaluated the performance of our scheme through simulations, and simulation results demonstrate that our scheme can timely wake up anchor sensors at a very low cost on energy consumption.

The rest of the paper is organized as follows. Section II discusses the related work. Section III describes the system model and the problems we addressed. Section IV gives the details of the wake-up scheme for train localization. Section V analyzes the energy efficiency of our scheme. Section VI presents the simulation results. Finally, Section VII concludes the paper and sheds some lights on future work.

II. Related Work

Existing wake-up schemes can be divided into two classes: synchronous wake-up and asynchronous wake-up. In synchronous wake-up protocols, sensor nodes wake up at the same time periodically to communicate with one another [5], [6]. Since all the participating nodes have to synchronize their clocks, synchronous duty-cycling is most appropriate for single-hop networks in which all the nodes can hear each another. The wake-up scheme presented in [7] requires some level of synchronization of clocks. The tracking scheme proposed in [8] is based on a combinatorics approach that sets delay bound at maximum target speed and ignores the timely tracking of object along with minimized energy consumption. However, real-time train localization is not delay tolerant due to the fast train speed. Also it is often difficult to predict at what time a train will pass by which sensor nodes, and thus it is impossible for synchronous duty-cycling protocols to use a static global schedule for all sensor nodes to wake up and to sleep. Moreover, it is nontrivial to synchronize distributed clocks of many sensor nodes [9].

In asynchronous duty-cycling protocols, sensor nodes are not required to synchronize their clocks with each other and sensor nodes can wake up independently. Since there are fewer communications among sensor nodes, asynchronous protocols are more energy efficient than synchronous protocols. Existing work on asynchronous wake-up schemes [10], [5] mainly focuses on the tradeoff between energy efficiency (i.e. network lifetime) and transmission latency. While our objective is to guarantee timely sensor wake up with the minimum energy consumption. Hence communication latency will affect the accuracy and reliability of localization and is not tolerable.

Other related works include a variety of MAC protocols designed based on asynchronous duty-cycling [11], [12], [13], [14]. Asynchronous duty-cycling provides a periodic channel sampling mechanism to detect potential transmissions. In order to start transmission, a sensor node transmits a long preamble packet to make it detectable by the neighbor nodes while each neighbor node performs CCA checks. A neighbor sensor node receives the preamble packet and prepares to receive data. Asynchronous duty-cycling protocols such as B-MAC [11], X-MAC [12] and Wise-Mac [13] deal with preamble packets in a way that the transmitter takes the responsibility to activate the receiver for data transmission. RI-MAC [14] eliminates the overhead of the preamble packet by letting the receivers initiate transmissions. However, these protocols are designed for general purpose and not suitable for train localization.

III. System Models and Problem Statement

A. Network Model

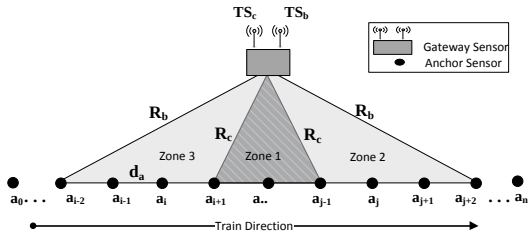

Fig. 1: *A WSN architecture for train localization*

The network consists of two types of sensor nodes: anchor sensor and gateway sensor, as shown in Figure 1. A set of anchor sensors $\{a_0, a_1, ..., a_n\}$ are uniformly deployed along a straight track with equal distance d_{a} between any two consecutive anchor nodes. Each anchor sensor is equipped with a single radio transceiver with transmission range of R_{c}. We assume that each anchor sensor is hard-coded its geographic coordinates before deployment. A single gateway sensor is installed on the train. The gateway sensor is equipped with two radio transceivers: TS_{c} and TS_{b}. TS_{c} is used to communicate with the anchor sensors that fall into its transmission range, and TS_{b} is used to continually broadcast beacon packets to activate the anchor sensors before they go into the transmission range of TS_{c}. The transmission range for TS_{c} and TS_{b} is R_c and R_b, respectively. We assume that R_b is larger than R_c. To avoid interference we assume TS_c and TS_b operate on two non-overlapping channels ch_c and ch_b respectively. Each anchor sensor operates on both channels, that is, uses ch_b during duty-cycling and switches to ch_c to communicate with TS_c. As shown in Figure 1, zone 1 is the region covered by TS_c, and zone 1, zone 2 and zone 3 are the region covered by TS_b.

The train localization scheme works as follows: as the train moves, TS_{b} continually broadcasts beacon packets. Each beacon packet contains information of the current train location (represented by the location of the gateway) and speed. Once an anchor sensor receives a beacon packet, it stops duty-cycling and switches to channel ch_c to prepare for communication with TS_c. When an anchor sensor goes into the transmission range of TS_{c}, it sends its geographic coordinates to the gateway sensor. After an anchor sensor finishes the communication with the gateway sensor, it switches back to channel ch_b and

resumes duty-cycling. Based on the geographic coordinates received from anchor sensors as well as the RSS information of the transmissions from anchor sensors, the train location will be computed at the gateway in a real-time manner.

B. Asynchronous Duty-Cycling Model

All anchor sensors operate in an asynchronous duty-cycling mode in which each anchor switches between sleep and wake-up states independently without global synchronization. Figure 2 shows one duty-cycle, in which an anchor sensor first sleeps for t_{sleep} second with its radio turned off, and then wakes up and turns its radio on to perform clear channel assessment (CCA) to detect incoming signals. If an incoming signal is detected, the anchor sensor will keep in active state until the scheduled communication between the anchor sensor and the gateway sensor is completed; otherwise it switches back to sleep state and repeats another duty-cycle. The length of one duty-cycle is represented by T_d, and the time for turning on/off radio and performing CCA is denoted by t_{sw} and t_{cca}, respectively.

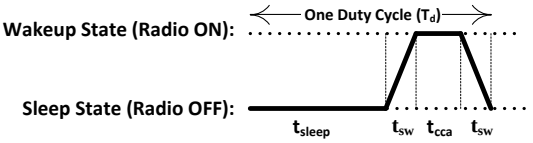

Fig. 2: *Illustration of one duty-cycle*

C. Problem Statement

In our train localization scheme, each anchor sensor must be in wake-up state and reports to the gateway once it goes into the transmission range of TS_c. However, each anchor sensor runs an asynchronous duty-cycling protocol, and can be woken up only if it detects the transmission signal from TS_b by performing CCA. The duty-cycling parameter t_{sleep} plays an important role in terms of timely waking up anchor sensors. If t_{sleep} is small, each anchor sensor needs to frequently turn on and turn off its radio, thereby wasting too much energy. From energy saving perspective, the larger the t_{sleep}, the more energy each anchor sensor can conserve. However, if t_{sleep} is too large, an anchor sensor may miss the chance to detect the beacon packet broadcast by TS_b and fail to wake up in time. The first issue we will address in this paper is to derive the upper bound on t_{sleep}, which enables that each anchor sensor can work in sleep state as long as possible while still guarantees that each anchor sensor can wake up in time once the train approaches.

The second issue we will address is to design an energy-efficient wake-up scheme, which guarantees that each anchor sensor can wake up in time once it goes into the transmission range of TS_c, and resume low power duty-cycling once it finishes communication with the gateway. The designed scheme will be evaluated through both theoretical analysis and simulations.

IV. BWS: Beacon-driven Wake-up Scheme

A. Upper Bound on t_{sleep}

As shown in Figure 3, suppose that anchor sensor a_i enters into the transmission range of TS_b at time t_b, and enters into the transmission range of TS_c at time t_c. Since anchor a_i must be active at time t_c to communicate with TS_c, it must wake up during the period $t_c - t_b$. To wake up, anchor a_i should receive at least one beacon from TS_b. Therefore, the following constraint on t_{sleep} has to be satisfied:

$$t_{sleep} \leqslant t_c - t_b, \tag{1}$$

otherwise, a_i may just start sleeping at t_b, and will still remain in sleep state at time t_c, thus will fail to wake up.

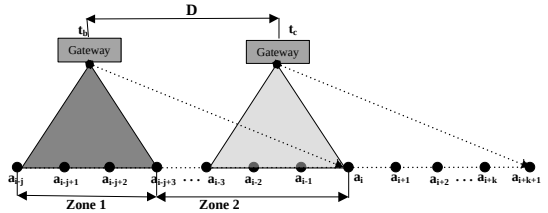

Fig. 3: *BWS: Illustration of Sensor-Train Communication*

Let D denote the distance travelled by the train during the period of $t_c - t_b$, and d_T represent the vertical distance from the gateway (train) to the line along which the anchor sensors are deployed. To find the upper bound for t_{sleep}, we first work out the size of Zone 2, as follows:

$$D_{z2} = \sqrt{R_{\mathrm{b}}^2 - d_{\mathrm{T}}^2} - \sqrt{R_{\mathrm{c}}^2 - d_{\mathrm{T}}^2}.$$

As d_T is very small as compared to R_b and R_c, this is ignorable and size of Zone 2 can be calculated as;

$$D_{z2} = R_{\mathrm{b}} - R_{\mathrm{c}}. \tag{2}$$

Let $S_{\max}$ represent the maximum train speed, at which the distance travelled by the train in the period $t_c - t_b$ is $D = S_{\max}(t_c - t_b)$. To guarantee that anchor a_i must perform CCA at least once in the period $t_c - t_b$ regardless the actual train speed, the following condition must be satisfied:

$$D_{z2} \geqslant S_{\max}(t_c - t_b) \tag{3}$$

Based on Equations (1), (2) and (3), we have

$$t_{sleep} \leqslant t_c - t_b \leqslant \frac{D_{z2}}{S_{\max}} = \frac{R_b - R_c}{S_{\max}}. \tag{4}$$

B. Design of BWS Protocol

The key idea behind the BWS protocol is to wake up any anchor sensor that goes into the transmission range of TS_b by detecting the beacon packets broadcast by the gateway. Specifically, the TS_b radio continuously broadcasts beacon messages that contain the following information: (a) the gateway ID (GW_ID), (b) the current train speed (S_T), and (c) the current train location (Loc_T). Once an anchor sensor receives a beacon packet from the gateway, it performs the following three tasks: *Duty-cycling Pause*, *communication with* TS_c and *Duty-cycling resumption*. In the Duty-cycling Pause task, the anchor sensor will pause the running of the default duty-cycling protocol and prepare for communication with TS_c. In the Communication with TS_c task, the anchor will report its location to the gateway via radio TS_C. Once the anchor moves out the transmission range of TS_c, it performs the third task to resume the default duty-cycling protocol. The three tasks are described in more detail below, and the pseudocode for wake-up protocol is given in Algorithm 1.

1) Duty-cycling Pause: Upon receiving of a packet, the anchor sensor first checks if the packet is a beacon packet broadcast by gateway (line 2 in Algorithm 1). As the received beacon packet contains the current train location, the anchor sensor can check in which zone it is located by comparing its location with the train location. If the anchor sensor is located in zone 2, it should first pause the running the duty-cycling protocol, stays active and then switches to channel from ch_b to ch_c (lines 3 and 4).

2) Communication with TS_c: When an anchor sensor a_i receives a beacon packet, it wakes up to prepare for communication with TS_c. The amount of time since a_i wakes up till it first goes into Zone 1 can be estimated by $\alpha = \frac{\sqrt{(x_{a_i}-x_T)^2+(y_{a_i}-y_T)^2}-R_c}{S_T}$, where (x_{a_i}, y_{a_i}) and (x_T, y_T) are the coordinates of a_i and the train respectively. Once a_i receives a beacon packet, it will initialized a timer, denoted by $start\_timer$ with value of α (line 7 in Algorithm 1). When $start\_timer$ expires, a_i will report its location to the gateway after getting medium clear by CSMA control layer (line 15 & 17). Once the gateway receives the packet from a_i, it will send back an acknowledgement. If a_i does not receive the acknowledgment, it will retransmit the packet after a short while (line 19 in Algorithm 1).

3) Duty-cycling resumption: Once an anchor node a_i goes out of the transmission range of TS_c (i.e., Zone 1 in Figure 1), it should resume the duty-cycling protocol. To achieve this, each anchor node maintains another timer called $stop\_timer$ (line 8). Once $stop\_timer$ is expired, the node should resume duty-cycling. The $stop\_timer$ is initialized with value β which is the amount of time elapsed since the node wakes up till the time it goes out of Zone 1, and β can be estimated using the train speed, that is, $\beta = \frac{\sqrt{(x_{a_i}-x_T)^2+(y_{a_i}-y_T)^2}+R_c}{S_T}$.

BWS enables an anchor sensor to accomplish these three tasks and guarantees the wake up of an anchor sensor for communication with TS_c. However, it is possible for an anchor sensor to get beacon packets when it is located in Zone 3 due to large omnidirectional transmission range of TS_b. BWS adaptively avoids unnecessary wake ups, but it uses the gateway's location information received in the beacon packet to calculate its zone. If an anchor sensor lies in Zone 3, BWS ignores such beacon packets and allows an anchor sensor to continue following duty-cycles. Although train location Loc_T may not be always accurate at a particular point of time, the associated localization error is acceptable by an anchor sensor to calculate its zone. However, for correct decision making for wake up, we assume that localization error due to time drift between anchor sensors and the train will never be larger than distance d_a.

Algorithm 1: Beacon-driven Wake-up at Anchor a_i

```
1  On receiving beacon packet:
2  if SourceID = GW_ID then
3      if a_i locates in Zone 2 then
                                        /* Pause duty-cycling */
4          duty_cycling = False
5          Channel = ch_c                /* channel switch */
6
7          start_timer ((sqrt((x_{a_i}-x_T)^2+(y_{a_i}-y_T)^2)-R_c)/S_T)
8          stop_timer ((sqrt((x_{a_i}-x_T)^2+(y_{a_i}-y_T)^2)+R_c)/S_T)
9      else
10         Ignore Beacon
11         duty_cycling = True
12 else
13     Ignore Beacon
                                        /* Resume duty-cycling */
14     duty_cycling = True
15 On start_timer expiry:
16 if start_timer is expired then
                                        /* Communicate with TS_c */
17     Send_Packet(x_{a_i}, y_{a_i})
18     Repeat process at line 17, until ack is received
19
20 On stop_timer expiry:
21 if stop_timer is expired then
22     Stop Sending Packets to Gateway
23     Channel = ch_b                    /* channel switch */
24
                                        /* Resume duty-cycling */
25     duty_cycling = True
```

States	Power Level	Energy Consumed
Transmission	P_{tx}	$E_{tx} = t_{tx}P_{tx}$
Idle Listening	P_l	$E_l = t_l P_l$
Packet Reception	P_{rx}	$E_{rx} = t_{rx}P_{rx}$
Switch radios b/w ON & OFF	P_{sw}	$E_{sw} = 2t_{tsw}P_{sw}$
CCA	P_{cca}	$E_{cca} = t_{cca}P_{cca}$
Sleeping	P_{sleep}	$E_{sleep} = t_{sleep}P_{sleep}$

TABLE I: *Anchor sensor's states and their power level*

V. Analysis of Energy Consumption

In our system, the energy consumed at each anchor sensor can be divided into two parts: energy consumed in duty-cycling and energy consumed in wake-up state. Table I gives the list of states in which an anchor sensor can operate and the corresponding power level for each state.

A. Energy consumed during wake up

If an anchor sensor goes to sleep at the point when it just goes into Zone 2, the amount of time that the anchor will sleep throughout Zone 2 is t_{sleep}. However, if the anchor sensor wakes up at the point when it just goes into Zone 2, it will receive a beacon packet and stay active. In this case the amount of sleeping time throughout Zone 2 is 0. Since duty-cycling is not synchronized among all anchor sensors, an anchor sensor may wake up at any time between the above two extremes when it is in Zone 2. The amount of time that an anchor sensor sleeps in Zone 2 follows a uniform random distribution between 0 and t_{sleep}. Hence the average amount of time that an anchor sensor stays in sleep state throughout Zone 2 is $t_{sleep}/2$.

The average amount of time that an anchor sensor stays in Zone 2 can be calculated by

$$\frac{R_b - R_c}{S_{avg}}, \tag{5}$$

where S_{avg} is the average train speed.

Let T_{z_2} denote the average amount of time that an anchor sensor stays active when it is in Zone 2 for one train pass. Then

$$T_{z_2} = \frac{R_b - R_c}{S_{avg}} - \frac{t_{sleep}}{2}. \tag{6}$$

Let T_{z_1} denote the average time that an anchor sensor stays in Zone 1 for one train pass. Then

$$T_{z_1} = \frac{2R_c}{S_{avg}}. \tag{7}$$

Let T_{wk} denote the average time that an anchor sensor stays in active state for one train pass. Since each anchor sensor will resume duty-cycling at point when it enters into Zone 3, we have

$$\begin{aligned} T_{wk} &= T_{z_1} + T_{z_2} \\ &= \frac{R_b + R_c}{S_{avg}} - \frac{t_{sleep}}{2}. \end{aligned} \tag{8}$$

To simplify our analysis we assume reliable communication between anchor sensors and the gateway. So each anchor sensor will receive one beacon packet, send one report packet and receive one ACK packet. Let t_{tx} and t_{rx} denote the time for transmitting and receiving a packet respectively. Therefore, the amount of time that an anchor sensor stays in idle listening state for one train pass, which is represented by t_l, can be computed as follows

$$t_l = T_{wk} - t_{tx} - 2t_{rx}, \tag{9}$$

Let E_{wk} denote the amount of energy consumed at an anchor sensor during wake-up state for one train pass. According to Table I ,

$$\begin{aligned} E_{wk} &= t_{tx}P_{tx} + 2t_{rx}P_{rx} + t_l P_l \\ &= t_{tx}P_{tx} + 2t_{rx}P_{rx} + (T_{wk} - t_{tx} - 2t_{rx})P_l. \end{aligned} \tag{10}$$

B. Energy consumed during duty-cycling

As shown in Figure 2, one duty-cycle includes three parts: sleep (t_{sleep}), CCA(t_{cca}) and state switch ($2t_{sw}$). Let E_{dc} denote the energy consumption for one duty-cycle. Then from Table I we have,

$$E_{dc} = 2t_{sw}P_{sw} + t_{cca}P_{cca} + t_{sleep}P_{sleep}. \tag{11}$$

The time required for switching radio between on and off states and the time for CCA check are constants, therefore the amount of energy consumed by state switching and CCA check is fixed for one duty-cycle. For simplicity, we use e_x to denote this amount of energy, that is,

$$e_x = 2t_{sw}P_{sw} + t_{cca}P_{cca}. \tag{12}$$

Then,

$$E_{dc} = e_x + t_{sleep}P_{sleep}. \tag{13}$$

C. Total energy consumption for a period

Let L be the total length of the time that the anchor sensors operate, and T_d be the length of one duty-cycle. We use λ to denote the total number of times that a train passes by an anchor sensor. Let E_{dc}^{total} be the total energy consumed during duty-cycling for the whole period L. Then

$$E_{dc}^{total} = \frac{L - \lambda T_{wk}}{T_d} E_{dc}, \tag{14}$$

where $\frac{L-\lambda T_{wk}}{T_d}$ is the total number of duty-cycles in time period L. Let E_{wk}^{total} be the total energy consumed during wake up for whole period L. Then

$$E_{wk}^{total} = \lambda E_{wk}. \tag{15}$$

Let E_L^{total} represent the total energy consumed by an anchor sensor in time period L. Then,

$$E_L^{total} = E_{dc}^{total} + E_{wk}^{total}. \tag{16}$$

Based on Equations (14) and (15), Equation (16) can be expressed as,

$$E_{total}^{L} = \frac{L - \lambda T_{wk}}{T_d} E_{dc} + \lambda E_{wk}. \tag{17}$$

By substituting Equations (8), (10) and (13) in Equation (17), we have,

$$\begin{aligned} E_L^{total} =& \frac{1}{2t_{sw} + t_{cca} + t_{sleep}} \Big(L - \lambda \Big(\frac{R_b + R_c}{S_{avg}} - \frac{t_{sleep}}{2} \Big) \Big) \\ & (e_x + t_{sleep}P_{sleep}) + \lambda \Big(t_{tx}P_{tx} + 2t_{rx}P_{rx} \\ & + \Big(\Big(\frac{R_b + R_c}{S_{avg}} - \frac{t_{sleep}}{2} \Big) - t_{tx} - 2t_{rx} \Big) P_l \Big) \end{aligned} \tag{18}$$

D. Optimal t_{sleep} for minimizing energy consumption

The minimization of energy consumed at each anchor sensor node can be formulated as the following optimization problem,

$$\begin{aligned} \text{minimize} \quad & E_{total}^{L} \\ \text{subject to} \quad & 0 < t_{sleep} \leqslant t_{sleep}^{ub} \end{aligned} \tag{19}$$

where t_{sleep}^{ub} is the upper bound for t_{sleep} which is given in Section III.A. As can be seen from Equation (18), the only variable is t_{sleep}, and it can be proved that E_{total}^{L} is strictly decreasing with the increase of t_{sleep}. The optimal t_{sleep} in terms of minimizing the total energy consumption at each anchor sensor is $t_{sleep}^{ub} = \frac{R_b - R_c}{S_{\max}}$.

VI. Simulation Results

A. Simulation Setup

We implement our scheme in TinyOS and run the simulations in the COOJA simulator [15]. In our simulations the train speed changes according to a random distribution between 5m/s and 10m/s, and we use the mobility plugin in COOJA to simulate the movement of the train. We modelled packet losses and retransmissions by using CSMA layer of CC2420 layer stack provided by TinyOS. The detailed configurations for the simulations parameters are given in Table II. As there are no specific wake-up schemes for the train localization, we have compared our simulational results with their theoretical counterparts.

Parameters	Values	Parameters	Values
Simulation time period L	1000s	R_b	$\sim$ 160m
Train trip frequency λ	1	R_c	$\sim$ 150m
No. of Anchor Sensors	800	Voltage	3.0v
Size of Zone-1	300m	Size of Zone-2 Zone-3	160m
d_T	2m	Size of Zone-3	160m
Simulation iterations	1000	Maximum train speed (S_{max})	10m/s
t_{sleep}^{ub}	16s	Minimum train speed	5m/s

TABLE II: *Simulation Parameters*

B. Number of Active Anchor Sensors in Zone 1

Since the size of Zone 1 is 300m and the distance between two adjacent anchor sensors is 100m, the number of anchor sensors that fall into Zone 1 can vary from 3 to 4. Figure 4 shows the number active anchor sensors in Zone 1 with different setting of t_{sleep}. According to Equation (4), the maximum t_{sleep} that can guarantee timely wake up of anchor sensors is 16 seconds. It can be seen that, for all case where t_{sleep} is not larger than 16s, the number of active anchor sensors that are active in Zone 1 fluctuates between 3 and 4. For the case where $t_{sleep} = 32s$, the number of active anchor sensors in Zone 1 varies from 0 to 4, and in most time there are only 1 or 2 active anchor sensors. This is because that the value of t_{sleep} (i.e., 32s) exceeds the upper bound t^{ub}_{sleep}. Since the

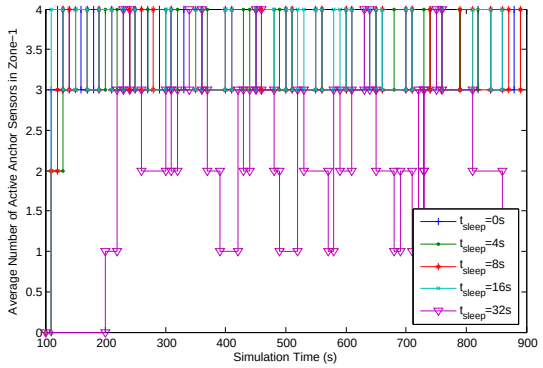

Fig. 4: *Number of active anchor sensors in Zone 1 at different* t_{sleep}.

size of Zone 1 is 300m, the anchor sensors are deployed with minimum density of $1sensor/100m$ ($\sigma = 4$ sensors in Zone 1) to maximum $8sensors/100m$ ($\sigma = 24$ sensors in Zone 1), the maximum number of anchor sensors within Zone 1 vary between total number of deployed anchor sensors in Zone 1 and one less than that (i.e., $\sigma = 24$, theoretically maximum anchor sensors in Zone 1 vary between 23 and 24). Even when all the anchor sensors follow the t^{ub}_{sleep}, the average number of active anchor sensors can be increased by densely deploying the anchor sensors along the track. Figure 5 shows that, for all the cases average number of active anchor sensors are increased as the deployment density (σ) is increased and the simulation results are very close to the theoretical counterparts.

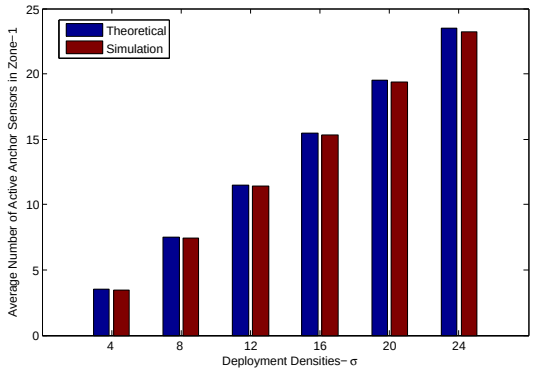

Fig. 5: *Number of active anchor sensors in Zone 1 at different deployment densities and* $t_{sleep} = 16s$.

However, if t_{sleep} is increased to a larger value (e.g. 32s) than t^{ub}_{sleep}, as shown in Figure 6, though the average number of active anchor sensors in Zone 1 increases along the increase in the deployment density, the average number of active anchor sensors decreases as compared to $t_{sleep} = 16$ (refer to Figure 5).

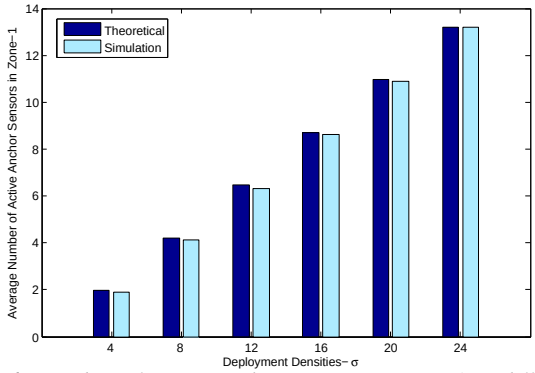

Fig. 6: *Number of active anchor sensors in Zone 1 at different deployment densities and* $t_{sleep} = 32s$.

C. Energy Consumption

Figure 7 shows the average energy consumption at each anchor sensor in simulation compared with their theoretical counterparts. All calculations are based on the current and voltage specifications of CC2420 radio chipset datasheet. It

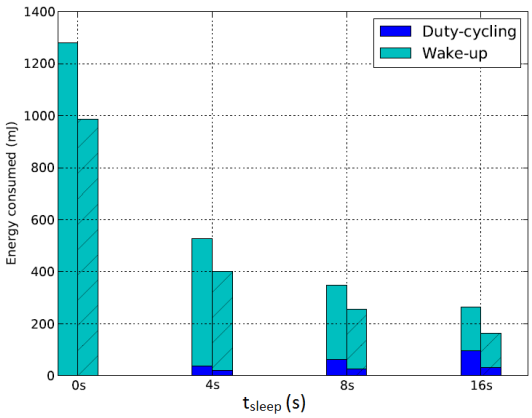

Fig. 7: *Energy consumed by an anchor sensor at different* t_{sleep} *values. Hatched bars are theoretical results, while clear bars are simulation results.*

can be seen that always keeping anchor sensors active without duty-cycling consumes a huge amount of energy. With the increase of t_{sleep}, the total energy consumption drops significantly because the energy consumed in active state drops. The amount of energy consumed by duty-cycling slightly increases since the amount of time that each anchor stays in sleep mode increases. Figure 7 also shows that the energy consumption obtained in simulations is close to the results computed based on our theoretical analysis. Note due to more realistic settings in the simulation (such as speed changes, packet loss and retransmission), more energy is consumed than in theory. Whereas, in theoretical results we didn't model packet losses, retransmission and only considered reliable transmission. However, it is obvious that the theoretical and simulation results match at the optimal sleep time (16s).

VII. Conclusions

This paper investigates the problem of designing an energy efficient communication protocol for real-time train localization using wireless sensor networks. In our scheme each anchor sensor runs an asynchronous duty-cycling protocol to conserve energy and wakes up only when it goes into the communication range of the gateway on the train. We designed a beacon-driven wake-up scheme and derive the upper bound on the anchor sensor sleep time within one duty cycle in order to guarantee timely wake up. We theoretically analyze our scheme on energy efficiency and evaluate its performance through simulations. Future work includes further improving our beacon-driven scheme, and evaluating possible schemes by conducting extensive simulation for comparison.

References

[1] A. Acharyaa, S. Sadhu, and T. Ghoshala, "Train localization and parting detection using data fusion," *Transportation Research Part C: Emerging Technologies*, vol. 19, pp. 75–84, 2011.

[2] M. Klepal, D. Pesch *et al.*, "A bayesian approach for rf-based indoor localisation," in *Wireless Communication Systems, 2007. ISWCS 2007. 4th International Symposium on*. IEEE, 2007, pp. 133–137.

[3] R. Vullers, R. van Schaijk, I. Doms, C. Van Hoof, and R. Mertens, "Micropower energy harvesting," *Solid-State Electronics*, vol. 53, no. 7, pp. 684–693, 2009.

[4] S. Beeby, M. Tudor, and N. White, "Energy harvesting vibration sources for microsystems applications," *Measurement science and technology*, vol. 17, no. 12, p. R175, 2006.

[5] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proc. of ACM/IEEE 7th International Conf. on Mobile Computing and Networking (MobiCom)*. ACM, 2001.

[6] A. Keshavarzian, H. Lee, and L. Venkatraman, "Wakeup scheduling in wireless sensor networks," in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2006, pp. 322–333.

[7] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3. IEEE, 2002, pp. 1567–1576.

[8] Y. Wong, L. Ngoh, W. Wong, and W. Seah, "A combinatorics-based wakeup scheme for target tracking in wireless sensor networks," in *Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE*. IEEE, 2007, pp. 3569–3574.

[9] R. Zheng, J. Hou, and L. Sha, "Asynchronous wakeup for ad hoc networks," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*. ACM, 2003, pp. 35–45.

[10] Y. Tseng, C. Hsu, and T. Hsieh, "Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks," in *Proc. of INFOCOM*, 2002.

[11] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 95–107.

[12] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, 2006, pp. 307–320.

[13] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: An ultra low power mac protocol for multi-hop wireless sensor networks," *Algorithmic Aspects of Wireless Sensor Networks*, pp. 18–31, 2004.

[14] Y. Sun, O. Gurewitz, and D. B. Johnson, "Ri-mac: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM, 2008, pp. 1–14.

[15] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Demo abstract: Cross-level simulation in cooja," in *Proceedings of the First IEEE International Workshop on Practical Issues in Building Sensor Network Applications*, 2006.