EAI Endorsed Transactions
on Future Internet
Research Article EAI.EU

MobiLab: A Testbed for Evaluating Mobility Management Protocols in Wireless Sensor Networks

Jianjun Wen, Zeeshan Ansar, Waltenegus Dargie*

Chair for Computer Networks, Faculty of Computer Science, Technical University of Dresden, 01062 Dresden, Germany

Abstract

Wireless sensor networks that support the mobility of nodes are finding applications in different areas such as healthcare, elderly care, and rehabilitation from total knee and hip replacement. However, these application areas also require reliable and high throughput networks. Considering the high fluctuation of link quality during mobility, protocols supporting mobile wireless sensor nodes should be rigorously tested to ensure that they produce predictable outcomes. In this paper we present a wireless sensor network testbed for carrying out repeated and reproducible experiments, independent of the application or protocol types which should be tested. The testbed consists of, among others, a server side control station and a client side traffic flow controller which coordinate inter- and intra-experiment activities. We fully implemented the testbed for the TinyOS and TelosB platforms. We employed Diddyborg robots for emulating different types of movement in indoor and outdoor environments. The paper includes also an extensive evaluation of the testbed and the performance of two mobility-aware MAC protocols.

Received on 17 November 2016; accepted on 6 July 2017; published on 21 December 2017

Keywords: Testbed, handover, mobility management, wireless sensor networks, mobile

doi:10.4108/XX.X.X.XX

1. Introduction

Wireless sensor networks which support the mobility of nodes are useful for different applications. For example, in the healthcare domain, they have been proposed to monitor patients with Parkinson Disease[1], gastroparesis [2], epilepsy [3], and asthma [4]. As a result, there is an endeavour to integrate medical devices and make them interact with existing wireless sensor platforms. For instance, the wireless mobility capsule integrating pH, pressure, and temperature sensors for the diagnosis of gastroparesis[1] has officially been approved by the US drug and food administration since 2006; it has produced promising results and may replace existing invasive and painful procedures (such as endoscopy) [5]. Similarly, there are commercially available wireless electrocardiograms which can be integrated with existing sensor platforms.

There are, however, some challenges associated with mobility, one of the most significant challenges being the difficulty of maintaining link quality during mobility. Independent experiments show that link quality quickly deteriorates when nodes are mobile while communicating, resulting in high packet loss, drift, and jitter. This aspect particularly affects applications which require relatively high throughput. Devices such as wireless electrocardiograms typically generate data at tens of kilobits per second rate. While this in itself may not be high, if other sensors such as 3D accelerometers and gyroscopes have to be sampled at comparatively the same rate, then the aggregate data rate from a single node can be high. Whereas the effect of mobility on link quality fluctuation has been extensively studied in the context of cellular communications (owing, luckily, to the ability of

*Corresponding author. Email: waltenegus.dargie@tu-dresden.de

[1] Gastroparesis is a condition in which the contraction of muscles in the stomach or intestine do not function properly, preventing the normal emptying of food in these organs. There can be many causes to gastroparesis such as uncontrolled diabetes, Parkinson Disease, multiple sclerosis, deposits of protein fibres in tissues and organs, and medicaments involving narcotics and antidepressants, but the primary cause is a damage to the vagus nerve, which regulates the digestive system. Its typical symptoms are nausea, vomiting, and constipation.

collecting ample statistics from a large number of users in different settings and locations), investigation of link quality fluctuation in mobile wireless sensor networks is a work in progress.

In this paper we propose a testbed for evaluating the effect of mobility in wireless sensor networks. The testbed separates the concern of application development from the evaluation of the application in different mobile scenarios. By doing so, complex and reproducible experiments can be carried out to ensure that the behaviours of applications are both reproducible and predictable. We fully implemented the testbed for the TinyOS and TelosB platforms. Our mobile nodes are carried by Diddyborg robots [6], each of which is controlled by 6 powerful gear motors, so that the robots can be tasked to emulate different types of movements in indoor as well as outdoor environments. We used our testbed to evaluate the performance of two proposed mobility management MAC protocols.

The remaining part of this paper is organized as follows: In Section 2, we review related work and position our own work. In Section 3, we present the system architecture of our testbed and in Section 4, we discuss its implementation. In Section 5, we employ our testbed to evaluate the performance of three independently developed mobility managing MAC protocols. Finally, in Section 6, we give concluding remarks and outline future work.

2. Related work

Testbeds are intended to efficiently test wireless sensor networks before they are actually deployed in real-world environments. Compared to the area or volume an actual deployment occupies, testbeds are considerably compact, so that they can be installed in labs or in areas which are easily accessible. This means, some communication parameters are intentionally scaled and events can be deliberately injected into the network to suit the test setting and to emulate actual events. There are online testbeds which are available to the WSN research community, most of them establishing two types of networks. One of the networks is the actual wireless link the characteristic of which is investigated and the other network serves as a backbone, reliable network for collecting performance indicator metrics. This network can be wireless (for example, a WLAN) or wired (using USB hubs or serial interfaces). As far as the software architecture is concerned, the existing testbeds also share similar aspects such as: (a) provision of web-based infrastructure and experiment management services; and (b) functionalities for dynamic reprogramming, specification, configuration, and execution of experiments. Some of the testbeds employ robots [7–9] while others employ toy trains [10] as mobile platforms, to which wireless sensor nodes are attached. Besides providing mobility, the mobile platforms also serve as power suppliers and node managers, through which new program images can be installed and experiment procedures are controlled and managed.

Emulab [7] is perhaps the first publicly reachable mobility-enabled testbed for WSNs experimentation. The testbed is deployed in an L-shaped area and consists of (1) 25 Mica2 static nodes installed on the walls and ceiling of a building to form a grid-like topology, (2) 6 mobile nodes attached to robotic platforms, which can perform user-specific and accurate way-point walking models (according to the authors, the position of the robots can be determined within 1 cm error, the worst-case), (3) 6 cameras which are installed on the ceiling to track the robots, and (4) additional 3 web-cams to provide live-monitoring. One of the limitations of the testbed is the difficulty of influencing the movements of the robots during experiment execution, because their movement pattern is predetermined and is not accessible at runtime.

Kansei [11] is a testbed employing the same types of robots like Emulab to support mobility, but it does not provide any positioning system. The testbed uses five robots integrating TMote Sky nodes and Extreme Scale Mote (XSM). These robots are deployed on top of a Plexiglas plane in which 210 XSMs and TMote Sky nodes are arranged in a 15×14 grid bench-work. In addition to the common functionalities the previous testbed provides, Kansei provides a mechanism to inject events into individual nodes and gateways. Sensei-UU [8] employs a Lego NXT robot as the mobile platform, on which a TelosB node and a smartphone are attached. Its unique feature is employing WL-500GP wireless access point as a control station to provide programming, experiment monitoring, and data logging functionality via a wireless channel. While it is relatively easy to reproduce and repeat experiments with this testbed, it has some drawbacks: (1) the robot requires the installation of tapes on the floor, which limits the types of movement that can be emitated by the mobile platform (i.e., undertaking different random movements is difficult); and 2) it is difficult to support multiple mobile nodes at the same time.

SensLAB [9] and TrainSense [10] are two recently proposed testbeds for mobile platforms. Both utilize toy trains as mobile platforms. Since the trains run on tracks, which physically limit their motion, the testbeds are difficult to extend. It is also difficult to introduce random walks into experiments. One of the merits of these testbeds is their ability to provide better accuracy of localization and control of mobility compared with the other mobile platforms.

3. Architecture of MobiLab

The main purpose and, therefore, contribution of our testbed is the flexible but reproducible execution of complex experiments with wireless sensor networks in which some of the nodes are mobile. The testbed enables users to upload their own program image onto individual nodes and to specify experiment procedures and the movement pattern of mobile robots independent of the types of applications the wireless sensor networks are supporting. To achieve these goals, our testbed separates resource management into different concerns.

3.1. HardwareArchitecture

The hardware architecture (displayed in Fig. 1) consists of four modules: a control station, a wireless sensor network, a node manager, and a backbone wireless channel. The control station serves as the main interface between the user and the testbed. A group of dedicated software services run in the control station to manage the testbed resources and to control experiments. In the next subsection we provide a detail description of the software architecture of the control station.

The wireless sensor network consists of three types of nodes: static relay nodes, mobile nodes, and sniffer nodes. The sniffer nodes are special stationary nodes that are not involved in any experiment, but are useful for monitoring the state of the wireless channel to obtain complementary information about experiment execution during debugging. By changing the firmware, the sniffer nodes can also produce interference into the network. A node manager interfaces a node with the control station. Each node manager (for our implementation we used a raspberry board) is connected to a wireless sensor node via a USB port. The node managers and the control station establish the backbone network to exchange management and experiment information at runtime. We use a Wi-Fi *ad hoc* network as our backbone network because of its scalability and flexibility. A node manager enables to easily program and control a node as well as to collect useful performance related data from it. The node manager connected to a mobile node has the additional task of controlling the motion of the robot and collecting location information.

3.2. Software Architecture

The control station is the most important module in MobiLab. It ensures that the testbed as a whole functions as a unified system. It is through the control station every program image or command is propagated to the wireless sensor network. Fig. 2 displays its software architecture, which consists of a user interface, a resource management service, an

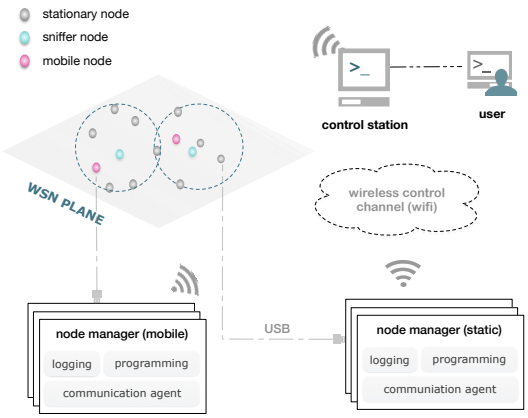

Figure 1. The hardware architecture of MobiLab.

experiment management service, a data management service, and a data analysis service.

User Interface. MobiLab provides both a web-based and a command-line-interface through which users can access the testbed and conduct experiments remotely. Users can browse active nodes and their status, upload program images into the wireless sensor network, and specify and manage experiment procedures using experiment execution primitives we defined (to be discussed below).

Rsource Management. MobiLab does not require a fixed infrastructure (a specific network size or topology) to run experiments. As to which specific pair of nodes should communicate with one another at any given time and for how long can be specified in experiment procedures to evaluate, for example, link quality fluctuation between them. The resource management service is responsible for authorizing nodes to join the network and users to access individual nodes; for managing binary images, and for ensuring proper program installation. Moreover, the resource management service uploads and deletes program images to and from nodes and controls versions. In it, a synchronization daemon runs in the background to ensure that program images in the control station and the node managers are consistent.

Experiment Management Service. The experiment management service enables users to define and manage inter- and intra-experiment activities. As regards management, users can initiate, interrupt, suspend, modify, and end experiments at runtime by using experiment execution primitives (see Table 1). The primitives enable users to configure interaction (transmission power, channel, partner nodes) and to specify communication durations, among others. When an experiment procedure is submitted to it, the experiment management service validates the procedure to ensure that it is executable, parse the procedure to extract experiment

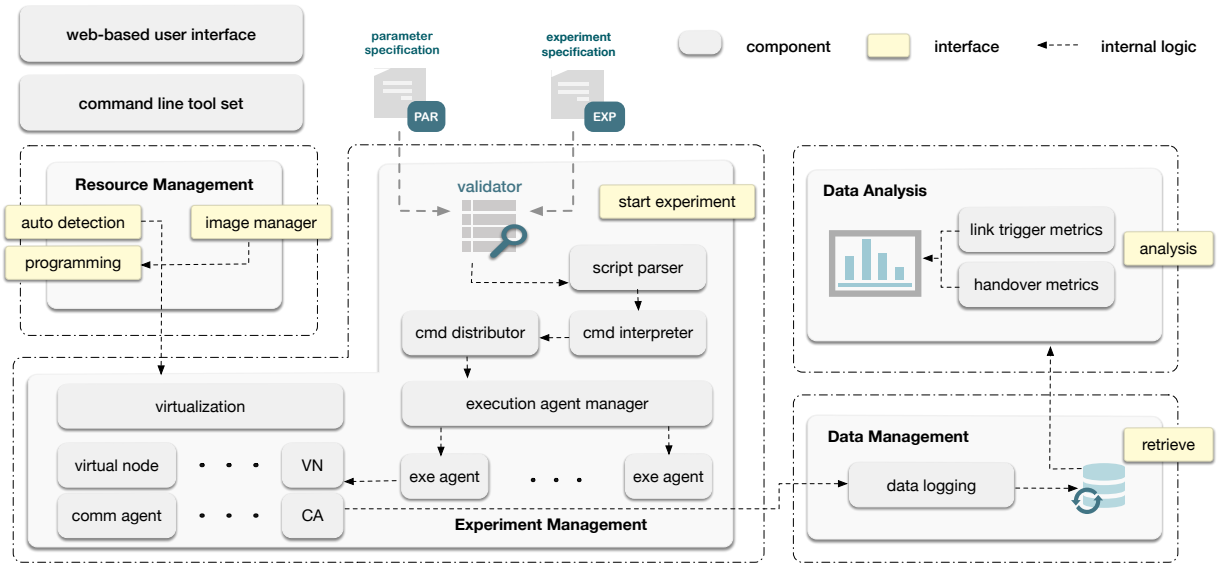

Figure 2. The software architecture of the control station.

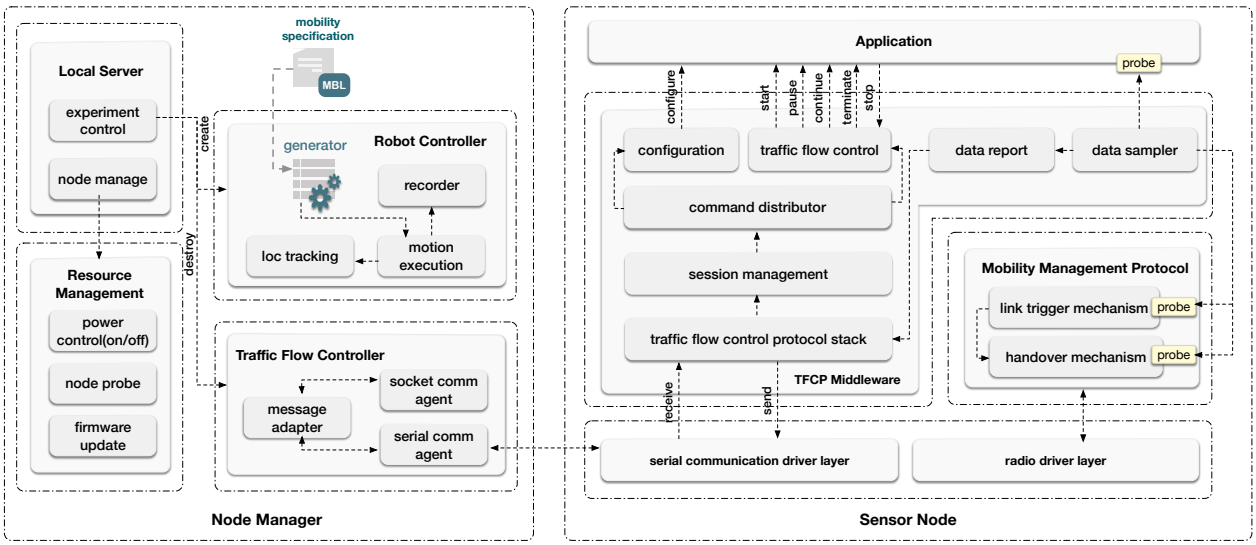

Figure 3. An overview of the software architecture of the testbed from a single node perspective: (Left) The software architecture of the node manager. (Right) The software architecture of a sensor node.

parameters, translates the parameters into binary, creates a control flow (execution sequence), and passes the control flow to the execution manager. The execution manager is responsible for coordinating the execution of an experiment procedure until it terminates. A virtual node manager within the control station's architecture creates a virtual representation for each physical node. The aim is to hide differences in hardware architecture between nodes from users and to provide common interfaces for accessing and interacting with them.

Table 1. traffic flow control primitives.

primitive	description
configure	setup the application dependent parameters
start	initiate the test round
stop	notify finish of test round
pause	suspend execution
continue	resume execution
terminate	stop execution permanently

Data Management and Analysis. Data management or logging is one of the useful features of testbed frameworks. The data in question are typically not

sensed data; rather they are performance indicators such as RSSI, Link Quality Indicator (LQI), SNR, timestamps, etc. which are useful for analyzing the fluctuation of link quality, the reliability of links, the adaptivity of routing protocols, etc. When an experiment is launched, MobiLab creates an instance of a data logging module which is then associated with the communication agents of the corresponding virtual nodes. During experiment execution, the physical nodes log the desired data locally and forward them to their virtual node managers at the control station, which then stores the data in a database. Alternatively performance indicators can be directly streamed to virtual node managers as they are generated.

3.3. Node Manager

A node manager is a physical device which is physically connected with a wireless sensor node via a USB interface. The idea is to facilitate the dynamic reprogramming of nodes, the replacement of modules, and the collection of relevant performance indicators during experiments. The software aspect of a node manager has three components, which are the local server, a resource manager, and a traffic flow controller. A robot node manager includes an extra module for managing mobility.

Local Server. It is a socket-based server that receives commands and messages destined to the physical node from the control station. Its main responsibility is managing the physical node and controlling the proper execution of experiments. The server is logically connected with the resource management service at the control station, thus it is able to provide the functionalities for probing the sensor node, updating firmware and physically powering on and off the node; it is also responsible for coordinating experiment control flows and commands pertaining to the motion of a robot.

Traffic Flow Controller. The procedure of an experiment is first encoded using the traffic primitives we specified in Table 1. By the time it reaches the traffic flow controller at the node manager, it is translated into a sequence of commands and parameters. The traffic flow controller is responsible for creating a channel between the node manager and the physical node and for transmitting the commands and parameters in their sequence and appropriate delay to the physical node. It also channels the logged data from the physical node to the node manager. The node managers are time synchronized with the control station at the beginning of each run of an experiment.

Robot Controller. The motion of a robot is controlled by a robot controller. The controller is instantiated by its node manager before an experiment is launched and destroyed after the experiment is completed. Different mobility models can be implemented and integrated into the node manager a priori and an instance of a model can be loaded when the robot controller is first instantiated. The parameters of this model can be modified at runtime by using the experiment primitives in Table 1. Currently, we are experimenting with straight line walking and the random waypoint model [12].

3.4. Sensor Node

Fig. 3 (right side) illustrates the software architecture of a wireless sensor node. Most relevant to this paper is the traffic flow control protocol middleware (TFCP), which we shall discuss in some detail. The TFCP middleware is an application independent layer for managing inter and intra-experiment activities. It is loosely coupled with the OS layer, interacting with communication drivers by `send` and `receive` interfaces and exposing six interfaces to the higher layers (MAC, network and application layers), so that users can setup experiment and application specific parameters and control the execution steps of experiments. The data sampler and report module locally collects and aggregates performance indicator metrics from relevant layers and communicates them with the control station via the TFCP middleware.

The mobility management protocol does not belong to the MobilLab testbed. We integrated it to investigate the performance of different mobility management protocols under the same setting. We shall explain this in more detail in Section 5.

4. Implementation

We established a wireless sensor network with TelosB nodes; its size varied between 10 and 20 stationary nodes and three mobile nodes. We implemented the control station on a laptop computer, in a Linux environment. Each TelosB node is physically attached to a Pi 2 model B raspberry board [13], which serves as a node manager. The raspberry board has 4 USB ports, a 900 MHz quad-core ARM Cortex-A7 CPU, 1 GB RAM, and 8 GB micro SD card. The mobile TelosB nodes are carried by Diddyborg robots [6] and controlled by their own onboard raspberry boards. The raspberry boards established a background ad-hoc network using USB-WiFi adapters and the ap-hotspot[2]. Several studies confirm that IEEE 802.11 and IEEE 802.15.4 radios can coexist with each other without obvious interference if non-overlapping channels can be selected carefully [14] [15], which we did.

[2]https://github.com/hotice/AP-Hotspot

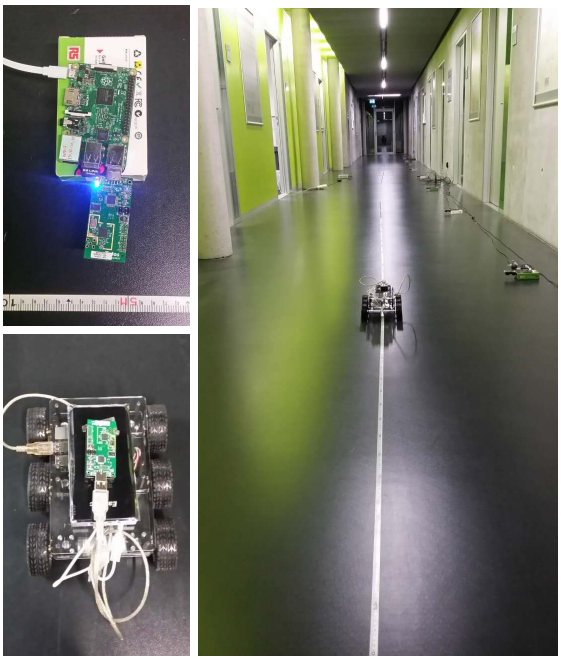

Figure 4. A wireless sensor network deployed in a long corridor at our faculty. One mobile node travels from one end of the corridor to the other end while communicating with a single relay node with which it establishes a link with the best link quality.

All the software components running inside the control station and the node managers are developed with Python, which can easily be ported to other platforms. The web application is based on Django Framework and Apache server. The TFCP middleware running on each sensor node is built on top of TinyOS and has a footprint of 1058 bytes of ROM and 84 bytes of RAM.

We re-implemented two mobility-aware MAC protocols, which we previously developed for MicaZ and Imote2 platforms [16–18] and integrated them into our testbed to evaluate both the testbed and the protocols. Both protocols are essentially the same and are intended to support burst transmission and seamless handover. Their main difference lies in their evaluation of link quality fluctuation and handover triggering mechanisms. The motivation for the protocols is that in residential or rehabilitation centers where mobile nodes can be employed to monitor patients, elderly people, or children, establishing reliable links is vital. Moreover, (1) the number of mobile nodes in these environments is small compared to the stationary relay nodes that can be deployed, (2) a significant portion of the network traffic is generated by the mobile nodes, and (3) the flow of traffic is predominantly one directional, namely, from the mobile nodes to the basestation. Consequently, these protocols forego contention for each packet to avoid unnecessary packet transmission delay; instead, as soon as a mobile node realizes that the channel is free, it transmits packets in burst, but each packet should be acknowledged to ensure reliable transmission. When the mobile node detects that the quality of an existing link is deteriorating, it searches for a more reliable relay node in its surrounding without breaking the existing link. Upon detecting a reliable relay node, the mobile node transfers communication to it. The key aspect of a handover is that a mobile node switches packet transmission from a unicast to a multicast mode during handover, so that neighbor relay nodes can intercept the packet it transmits and candidate themselves to become its relay nodes. From candidate relay nodes, the mobile node chooses one of them and switches communication back to a unicast mode.

The link quality evaluation mechanisms and whether a deterioration in a link quality eventually leads to a disconnection require the evaluation of incoming acknowledgement packets. A simpler mechanism leads to a quicker decision, introducing a small processing overhead but the decision may also lead to frequent handover oscillation; a more complex mechanism requires a sizable amount of received acknowledgement packets and a more advanced estimation technique. The first protocol can be configured as *single-threshold* or *dual-threshold*. In the first case, when the RSSI values of a set number of incoming acknowledgement packets consistently drops below a set threshold, a handover is initiated. In the second case, two types of thresholds are defined. The first threshold servers to initiate a handover request whereas the second threshold servers to select and bind to a new relay node. The second protocol, on the other hand employs an adaptive filter (least mean square) to determine link quality deterioration as a consistent phenomenon and to initiate a handover.

5. Evaluation

As we already point out, we used our setup to evaluate both the testbed as an experiment supporting tool and the mobility-aware MAC protocols. As regards the testbed, our aim was to evaluate the variance in the experiment dissemination and completion time, the scalability of the testbed both as a function of the number of nodes involved in an experiment and as a function of the complexity of an experiment. As regards the MAC protocols, our aim was to evaluate the cost of handover, in terms of the expected delay in transferring a communication as a function of node density (distance between relay nodes) and the duplication of packets during a search for a relay node (the mobile node communicates with multiple relay nodes during a handover request or neighbor discovery phase).

5.1. Evaluation of MobiLab

In order to carry out reproducible experiments, the detail of the experiment procedures are scripted, i.e., the beginning, end, and duration of every activity is specified. When the control station dispatches experiment procedures, they may not be executed by the individual nodes at precisely the same time. Consequently, nodes may not begin and end the execution of experiments at the same time. This phenomenon is an aspect of both the size of the network and the complexity of the experiments.

To investigate this phenomenon, we launched a set of simple experiments with variable number of physical nodes (from 5 to 20) and emulated nodes (up to 500). We recorded the starting time of each node and calculated the maximum variance (time difference between the earliest starting node and the latest staring node). We observed that the maximum variance of experiment beginning time was 200 ms. Secondly, we inserted arbitrary number of control commands (*pause* and *continue* commands) in the experiments lasting up to 600 seconds, and varied the number of nodes from 1 to 10. We did not observe significant increments of experiment completion times when the number of commands increased (shown in Fig. 5b). To show the repeatability of an experiment using MobiLab, we evaluated the RSSI fluctuation and packet reception rate under different configurations. We conducted a series of experiments and repeated each one three times. As Fig. 5c and 5d shown, the CDF of RSSI values are almost the same for the experiment with the same configuration (transmission power, motion pattern, walking path etc.) and the packet reception rates are consistent for each repetition.

5.2. Evaluation of Mobility-Aware Protocols

We carried out repeated experiments to evaluate the performance of the two mobility-aware protocols in terms of overall packet transmission success rate and the number of duplicate packets relay nodes should forward to the basestation during a handover phase. Our experiments involved two different separation distances between relay nodes that make up the wireless sensor network we deployed in one of the corridors of our faculty: 5 m and 10 m (see Fig. 4). A mobile robot carrying one of the mobile nodes moved at a speed of 0.13 m/s from one end of the corridor to the other end while transmitting packets in burst. It needed about 200 seconds to cover the entire distance. For each configuration we repeated the experiment three times. Table 2 lists the physical and MAC layer parameters for out experiments.

Success Rate. Fig. 6a displays the packet transmission success rate of the thee handover schemes. As can be seen, all of them have a success rate of greater than 98% when the distance of separation between the relay nodes was 5 m, for the overlapping in the radio coverage of the relay nodes becomes larger when the separation distance becomes smaller, as a result, a handover request is most likely successful. In contrast, when the separation distance between the relay nodes was 10 m, the packet transmission success rate of all the three schemes was reduced; nevertheless, the LMS scheme outperformed the other two schemes by a narrow margin (98.33% compared with 94.92% and 96.67%).

Table 2. experiment parameters: upper) link trigger independent parameters; lower) link trigger dependent parameters

parameter		value
deployment (spacing)		5, 10 m
channel		26
transmission power		-25 dBm
IPI		200, 500 ms
duration		200 s
link trigger	**parameter**	**value**
Single threshold	threshold	-60, -65 dBm
Dual threshold	threshold	-60, -65 dBm
	margin	5 dBm
LMS	partitions	5
	trigger level	2

Handover Cost. In order for a mobile node to search for an alternative relay node in its surrounding without interrupting communication with the old relay node, it has to transmit outgoing packets in a broadcast mode. These packets will be intercepted by participating relay nodes and forwarded to the basestation, which leads to packet duplication. The faster the handover phase, the fewer the duplication packets. The slower the handover phase, the larger the duplication packets. We defined the term duplication ratio to quantify the networks channel utilization efficiency. Thus,

$$\delta = \frac{p_\delta}{p_u} \tag{1}$$

where P_δ and P_u are the number of duplicated and unique packets the basestation receives, respectively, and δ is a measure of channel utilization efficiency (a larger δ signifying a lower channel utilization). Fig. 6b displays the network's channel utilization efficiency in terms of δ. As can be seen, when the distance of separation between the relay nodes was 5 m, the duplication ratio of the *single threshold* and the *dual threshold* handover strategies was significantly higher. This is the trade-off between a higher handover success rate and a larger amount of duplication. As the separation distance between relay nodes decreases, the chance of nearby relay nodes intercepting the

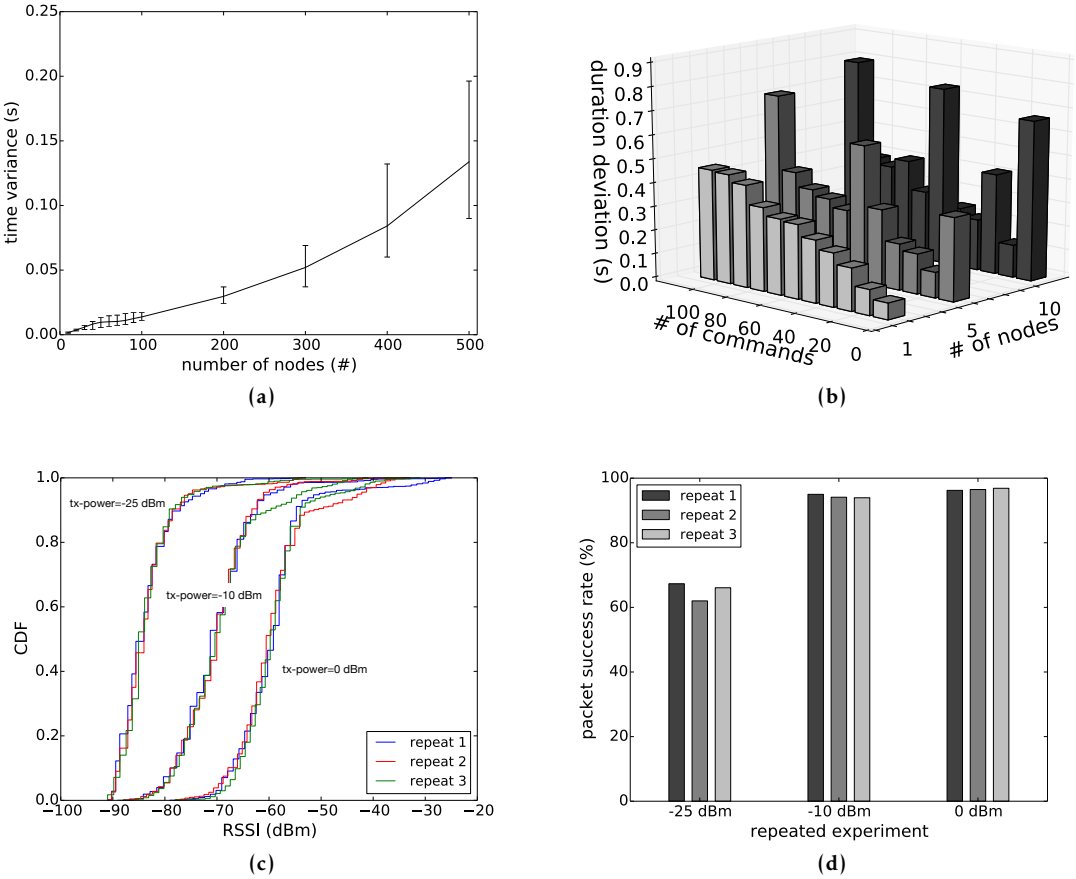

Figure 5. Performance analysis of MobiLab as an experiment management tool: (a) The time variance of synchronized starting time for networks of different sizes; (b) The deviation in the duration of arbitrary control commands in a single experiment. (c) cdf of RSSI fluctuation of repeated experiments; (d) packet success rate of repeated experiments

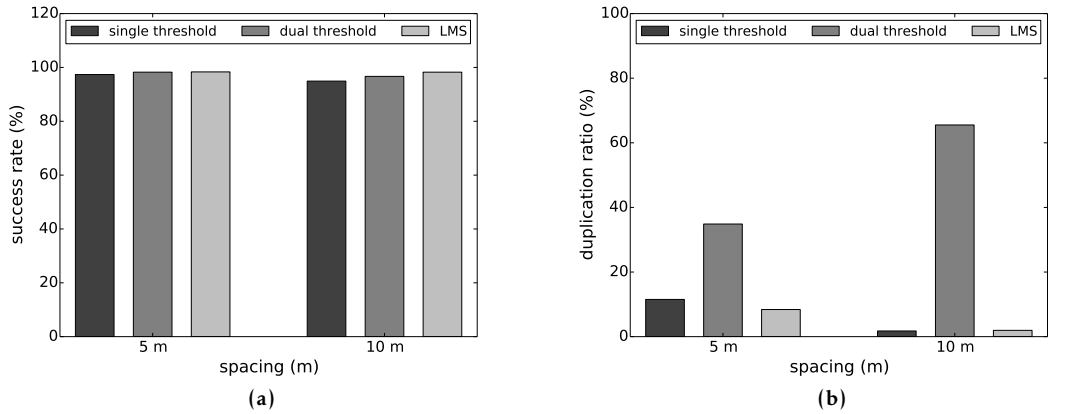

Figure 6. The efficiency of handover mechanisms for different deployment scenarios (when the spacing between stationary relay nodes was 5 m and 10 m). (a) Packet transmission success rate; (b) The duplication ratio of the whole network.

multicast handover packets increases. The relay nodes intercepting the handover request packets (the payload of which contains a useful data) will forward the packet to the basestation (in order to reduce packet loss due

to the deteriorating link quality). At a longer distance of separation, the chance of intercepting a handover request decreases, but with it, the possibility of finding an alternative relay node decreases as well. As can be seen in the figure, there is an exception to this for the *dual threshold* scheme; this is because, this scheme forces the mobile node to wait until it discovers a link, the quality of which is above the second threshold, before it finalized a handover process.

Handover Oscillation. Fig. 7 shows how the three handover schemes enabled a seamless handover as a result of fluctuation in the RSSI values of received acknowledgement packets (as regarded from the mobile node, which initiates a handover). The left-side of each figure indicates the RSSI values in dBm and the right-side indicates in which of the two states – a normal transmission state (unicast transmission) or a handover state (multicast transmission state) – the mobile node was while moving along the corridor. The dual-threshold scheme spent much of the time in searching for reliable relay nodes whereas the single threshold and the LMS schemes spent comparatively much of the time in a normal transmission state (signifying a quick handover phase). However, the LSM scheme is the one with the highest successful packet transmission rate.

6. Conclusion and Future Work

In this paper we introduced MobiLab, a testbed we developed to experiment with wireless sensor networks which support mobile nodes. MobiLab separates the concerns of application and protocol developments from their testing phase. Our main motivation was performing repeated and reproducible experiments independent of the types of network topology, communication protocols, communication parameters, and sensors involved in the experiments. We presented both the conceptual architecture and the implementation of our testbed and illustrated how we used the testbed to evaluate two mobility-aware MAC protocols. The hardware architecture of MobiLab consists of a control station, node managers, sensor nodes and mobile robots as mobile platforms. The node managers establish a wi-fi backbone network to provide management and data collection functionalities during the execution of experiments. The wireless network enables flexible deployment and scalability. The hardware components of MobiLab are commercial off-the-shelf (COTS) products, which makes MobiLab affordable and easy to reproduce. From the software perspective, besides sharing the same design principles with existing testbeds, MobiLab provides several novel contributions such as supporting both inter- and intra-experiment management, TFCP middleware in a sensor node, and a robot motion management. Except for the sensor node architecture, which is implemented in C for the TelosB platform, all the remaining software components are implemented in python, which is relatively easy to port to other platforms. Our future goal is to use the testbed for testing different mobile applications and routing and MAC protocols, to enlarge the wireless sensor network, and to deploy the testbed in different environments.

Acknowledgement. This work has been partially funded by the German Research Foundation (DFG) under project agreement: DA 1211/5-2.

References

[1] Sha, D.L., Xie, W.C., Fan, X.L. and Li, Y. (2015) Based on wireless sensor network (nwk) of non-contact tremor monitoring equipment improvement for parkinson's disease. In *Applied Mechanics and Materials* (Trans Tech Publ), **713**: 491–494.

[2] Grimes, C.A., Ong, K.G., Varghese, O.K., Yang, X., Mor, G., Paulose, M., Dickey, E.C. *et al.* (2003) A sentinel sensor network for hydrogen sensing. *Sensors* **3**(3): 69–82.

[3] Lay-Ekuakille, A., Vergallo, P., Griffo, G., Kanoun, O., Angelillo, F. and Trabacca, A. (2013) Wireless network acquisition of joint eeg-ecg-ergospirometric signals for epilepsy detection. In *Measurements and Networking Proceedings (M&N), 2013 IEEE International Workshop on* (IEEE): 41–45.

[4] Seto, E.Y., Giani, A., Shia, V., Wang, C., Yan, P., Yang, A.Y., Jerrett, M. *et al.* (2009) A wireless body sensor network for the prevention and management of asthma. In *Industrial Embedded Systems, 2009. SIES'09. IEEE International Symposium on* (IEEE): 120–123.

[5] Farmer, A.D., Scott, S.M. and Hobson, A.R. (2013) Gastrointestinal motility revisited: the wireless motility capsule. *United European gastroenterology journal* : 2050640613510161.

[6] Diddyborg robot, https://www.piborg.org/diddyborg.

[7] Fish, R., Flickinger, M. and Lepreau, J. (2006) Mobile emulab: A robotic wireless and sensor network testbed. In *IEEE INFOCOM*.

[8] Rensfelt, O., Hermans, F., Larzon, L.Å. and Gunningberg, P. (2010) Sensei-uu: A relocatable sensor network testbed. In *Proceedings of the fifth ACM international workshop on Wireless network testbeds, experimental evaluation and characterization* (ACM): 63–70.

[9] Des Rosiers, C.B., Chelius, G., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N. and Noël, T. (2011) Senslab very large scale open wireless sensor network testbed. In *Proc. 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCOM)*.

[10] Smeets, H., Shih, C.Y., Zuniga, M., Hagemeier, T. and Marrón, P.J. (2013) Trainsense: a novel infrastructure to support mobility in wireless sensor networks. In *Wireless Sensor Networks* (Springer), 18–33.

[11] Ertin, E., Arora, A., Ramnath, R., Naik, V., Bapat, S., Kulathumani, V., Sridharan, M. *et al.* (2006) Kansei:

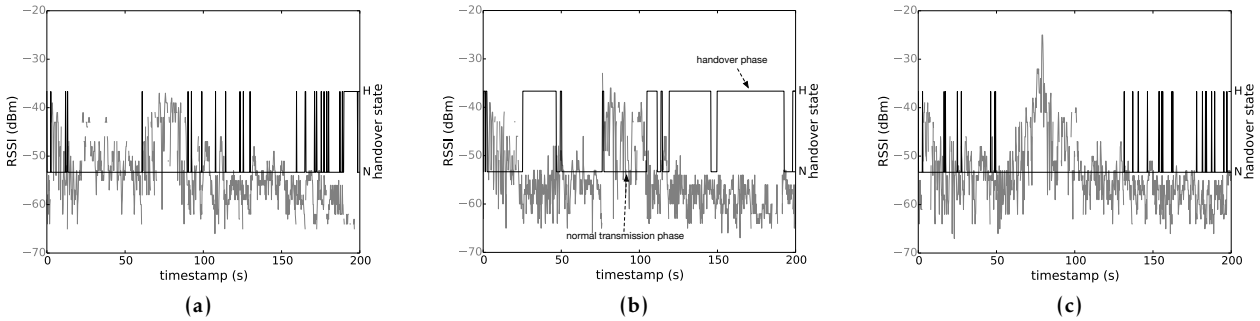

Figure 7. RSSI fluctuation of incoming acknowledgement packets and handover oscillation. (a) Single threshold scheme. (b) Dual threshold scheme. (c) LMS.

a testbed for sensing at scale. In *Proceedings of the 5th international conference on Information processing in sensor networks* (ACM): 399–406.

[12] Camp, T., Boleng, J. and Davies, V. (2002) A survey of mobility models for ad hoc network research. *Wireless communications and mobile computing* **2**(5): 483–502.

[13] Raspberry pi 2 model b, https://www.raspberrypi.org/products/raspberry-pi-2-model-b/.

[14] Srinivasan, K., Dutta, P., Tavakoli, A. and Levis, P. (2010) An empirical study of low-power wireless. *ACM Transactions on Sensor Networks (TOSN)* **6**(2): 16.

[15] Baccour, N., Koubaa, A., Mottola, L., Zuniga, M.A., Youssef, H., Boano, C.A. and Alves, M. (2012) Radio link quality estimation in wireless sensor networks: a survey. *ACM Transactions on Sensor Networks (TOSN)* **8**(4): 34.

[16] Dargie, W. and Wen, J. (2014) A seamless handover for wsn using lms filter. In *Local Computer Networks (LCN), 2014 IEEE 39th Conference on* (IEEE): 442–445.

[17] Dong, Q. and Dargie, W. (2013) A survey on mobility and mobility-aware mac protocols in wireless sensor networks. *Communications surveys & tutorials, IEEE* **15**(1): 88–100.

[18] Zhiyong, T. and Dargie, W. (2010) A mobility-aware medium access control protocol for wireless sensor networks. In *GLOBECOM Workshops (GC Wkshps), 2010 IEEE* (IEEE): 109–114.