Talk to Me! On Interacting with Wireless Sensor Nodes

Matthias Gauger
Universität Bonn, Bonn, Germany and
Fraunhofer IAIS, St. Augustin, Germany
gauger@cs.uni-bonn.de

Olga Saukh
Structural Engineering Research
Laboratory, EMPA, Switzerland
olga.saukh@empa.ch

Pedro José Marrón
Universität Bonn, Bonn, Germany and
Fraunhofer IAIS, St. Augustin, Germany
pjmarron@cs.uni-bonn.de

***Abstract*—Wireless sensor networks play an essential role in many pervasive computing scenarios as providers of context data. However, interacting with sensor nodes and selecting specific nodes for an interaction is difficult due to the constraints of the sensor node hardware. In this paper, we present and discuss three different approaches for this node interaction problem based on gestures, on light signals and on information provided by the sensor nodes using their LEDs. We demonstrate in our evaluation that these mechanisms solve the interaction problem for a variety of scenarios and effectively support the user in the interaction process. At the same time, our solutions set only low requirements on the wireless sensor node hardware.**

I. Introduction

Individual sensor nodes and networks of sensor nodes form an integral part of many pervasive computing scenarios as providers of context data. However, the integration of these extremely resource-poor devices poses several unique challenges that require special solutions. One such challenge is the interaction of mobile devices with sensor nodes. While sensor nodes are often considered as simple information providers that only send information to a single base station node, in pervasive computing scenarios we also perceive the need for direct interactions between mobile devices and sensor nodes.

Establishing an interaction is particularly difficult due to the properties of the wireless sensor nodes. They do not possess any of the conventional user interface components for inputting and outputting information like a keyboard or a display. Instead, sensor nodes usually interact with neighboring devices solely by exchanging messages using the wireless communication interface.

In this paper we consider the problem of initiating a purposeful interaction between mobile client devices and resource-constrained wireless sensor nodes. We explore different ways of interacting with individual wireless sensor nodes or groups of nodes. In particular, we investigate how a user can be enabled to interact with specific sensor devices without requiring pre-existing knowledge on the properties of the nodes (e.g., the node identifiers) or the structure of the network (e.g., the node positions). With gesture-based interaction, Sensor Node Lamp interaction and *NextOnePlease* interaction we propose three solutions to the problem of initiating such an interaction that build on different assumptions concerning the capabilities and tasks of the sensor nodes on the one side and the user application and the users on the other side.

In the context of wireless sensor networks, we aim for several types of applications that go beyond querying sensor data from a specific node. One example is to use interactions as part of deploying and configuring a sensor network, e.g., to assign roles to nodes or to perform selective code updates. Interaction capabilities can also be used for sensor network debugging, for example to query the state of specific nodes or to determine which nodes need to be exchanged. In addition to such sensor network-specific application ideas, the ability to associate mobile users with devices and services embedded in the environment is an important problem in wireless networks and pervasive computing in general and the approaches described in this paper might be applicable there as well.

The rest of this paper is organized as follows. The following section discusses approaches related to our work before Section III introduces our problem definition and the goals of this work. We discuss our three different methods for interacting with sensor nodes in Sections IV, V and VI. We then present the evaluation of our approach in Section VII. Finally, Section VIII concludes the paper and outlines future work.

II. Related Work

Techniques for mobile interaction between users and embedded devices are an important research topic in mobile and pervasive computing. In this section, we discuss the fundamental interaction techniques touching, pointing and scanning to set our solutions into context. We also consider existing solutions developed specifically for wireless sensor networks.

A. Touching, Pointing and Scanning

Our gesture-based interaction technique can be considered a member of the group of **touching** interaction techniques that require the user to walk up to a device to select it for interaction. Other representatives of this group work based on RFID tags and readers, proximity sensors or user buttons on the target device (e.g., [1], [2], [3]). Gesture recognition systems, in contrast, are typically used as input mechanisms in the context of mobile, wearable and pervasive computing (e.g., [4], [5], [6]).

Pointing interaction techniques require the user to point a mobile device in the direction of the node they want to interact with. In previous work, this has been implemented with cameras that read visual codes attached to objects or that

978-1-4244-3304-9/09/$25.00 ©2009 IEEE

perform object recognition (e.g., [7]). An alternative approach works with the help of directed infrared signals sent by the objects of interest [8]. Our Sensor Node Lamp mechanism belongs to the group of techniques that use light signals for the pointing action. Ghali et al. [9] do this by tracking the light cone of a flashlight with a camera. Ma and Paradiso [10] describe a system using a modified laser pointer that transmits a high-frequency optical signal to custom-built tags that are able to filter out interferences and use the data signal to detect whether they are target of the interaction. Rukzio et al. [2] use a similar approach receiving light from a laser pointer with an array of light sensors.

The idea of **scanning** interaction techniques is to collect information on smart devices in the vicinity of the user on the user's client device. This information is then presented to the user in appropriate form to allow him to select objects for interaction. The collection of information is usually done using radio communication [2]. Our approach *NextOnePlease* also lets the user select the target node for an interaction from the set of neighboring nodes. However, it differs from scanning interaction techniques in that it cannot rely on the sensor nodes describing themselves sufficiently and therefore works with additional visual feedback.

Rukzio et al. [2] investigated the mobile interaction techniques touching, pointing and scanning concentrating on the preferences of users. They found that people generally prefer touching for nearby objects and pointing for objects farther away. Scanning is only preferred for objects outside of the user's line of sight.

The selective interaction with devices also plays an important role in the context of security, namely for authentication. Smart-Its Friends [11] connects ("peers") two devices when a user holds them together and shakes them together. This way, the explicit intention of the user to peer these devices is easily checked. Patel, Pierce and Abowd [12] describe a system that verifies that a user wants to release data from his mobile device to a public terminal by displaying a gesture challenge on the public terminal. The user must confirm his intention by shaking the device as specified on the display.

B. Interacting with Sensor Nodes

There are relatively few papers so far that specifically consider the interaction with wireless sensor nodes. Peterson and Rus [13] describe a device they call Sensory Flashlight that allows to interact with sensor nodes lying in a specific direction of the user. They assume that both sensor nodes and the Sensory Flashlight know their exact positions. Using this location information together with directional information from an analog compass, they are able to determine which devices the Sensory Flashlight is pointing at.

The Tricorder [14] realizes the selective interaction with sensor nodes in an indoor environment using location and direction information. The Tricorder determines its orientation using an integrated compass and its approximate location using a simple RSSI-based localization mechanism with the static sensor nodes acting as location beacons. Sensor nodes can then be selected on the display of the Tricorder showing the nodes in a map of the building. Unlike both the Sensory Flashlight and the Tricorder, our approaches work without relying on location information.

MoteFinder [15] uses a special directed antenna (a so-called cantenna) to strengthen the reception of radio messages from a certain direction while attenuating messages from other directions. The RSSI values of received beacon messages can then help to identify a specific node by pointing the cantenna in its direction. However, signal propagation effects limit the applicability of MoteFinder, particularly in indoor scenarios.

III. Problem Definition

There are various types of interactions possible for a mobile device operating in the area of a wireless sensor network. Besides interacting with all neighboring nodes (**broadcast interaction**) or with a single, randomly selected node (**random interaction**), we are particularly interested in interactions with a specific node (**individual interaction**) or a group of nodes (**group interaction**) that the user selects based on real-world criteria. Selecting all nodes or a single random node for interaction is trivial to implement, the first using a simple broadcast, the second by selecting any node in the neighborhood. For that reason, we concentrate on individual and group interaction in the following discussion.

We assume that the user carries a mobile device, called client node, that he uses for interacting with his environment. While the user is usually able to informally describe which node or which set of nodes he targets (e.g., by describing its position: "I want to query the node mounted on the wall to my left"), it is often difficult or impossible to specify the identifiers of the destination nodes which would be required to establish a one-to-one communication between the client node and the destination node. At the same time, interaction messages sent by broadcast might reach a whole set of candidate nodes. The problem of performing a targeted, purposeful interaction is to select the set of destination nodes from the set of candidate nodes while avoiding to also select any undesired node.

The primary goal for our interaction mechanisms must be a **high accuracy** in the selection of nodes. The occurrence of both false positives (i.e., selecting nodes that the user did not intend to interact with) and false negatives (i.e., ignoring nodes chosen by the user) should be avoided. Closely related to that is our second goal, a **good usability of the system**. It must be simple and intuitive to use – despite the limitations of the sensor nodes involved in the interaction. Finally, we also need to take the resource limitations of the wireless sensor nodes into account. For that reason, we need to **limit the complexity** of the operations performed on the individual nodes. Moreover, we cannot require additional hardware (e.g., a display) on the sensor nodes just for facilitating the interaction.

The main target scenario for our prototype implementation are indoor deployments of wireless sensor networks like in homes, office buildings or factories. However, some of the presented solutions are also applicable in outdoor settings.

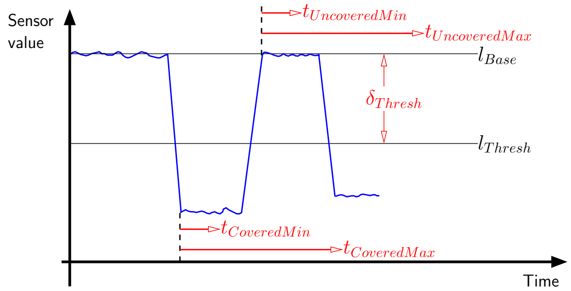

Fig. 1. Illustrated gesture recognition process

In the following sections, we present three different ways of realizing interaction with wireless sensor nodes that concentrate on different aspects and solution approaches: **Gesture-based interaction**, the **Sensor Node Lamp interaction** approach and the LED-based mechanism ***NextOnePlease***. As we will show, these three approaches allow us to cover a wide range of different scenarios and types of interaction setup.

IV. Gesture-Based Interaction

The idea of gesture-based interaction is to define a simple gesture that the user is able to perform in front of a sensor node using one of his hands and that can be detected solely by analyzing variations in the recorded light level over time using the integrated light sensor. A node is then selected for interaction if it detects such a gesture performed by the user.

A. Approach

We detect gestures performed by the user moving his hand over the sensor node. Such a movement results in an uncovered-covered-uncovered sequence of states that can be detected using the light sensor values recorded during that time. Due to varying speeds of the hand movement, we have to expect differences in the length of both the "sensor covered" and the "sensor uncovered" time periods in between and cannot set tight bounds in the recognition algorithm.

The gesture recognition starts with a base sensor value l_{Base} and requires to go below a threshold sensor value $l_{Thresh} = l_{Base} - \delta_{Thresh}$ to detect a transition to the "sensor covered" state. Fig. 1 illustrates the threshold and timeout values used as part of the gesture recognition process including minimum and maximum lengths of the "sensor covered" and "sensor uncovered" time periods ($t_{(Un)CoveredMin/Max}$).

While larger values for δ_{Thresh} decrease the likelihood of false positive events caused by normal light level variations, they can also hinder the detection of gestures, particularly in dimly lit scenarios. To account for this and the influence of varying external conditions, the sensor nodes determine the value of l_{Thresh} dynamically in a self-calibration process. We found that – while the absolute light levels change – the ratio $\frac{l_{Base}}{l_{Covered}}$ remains more or less constant. We therefore calculate the threshold value as follows: $l_{Thresh} = a \times l_{Base}$ with $0 < a < 1$. Smaller values of a reduce the probability of false positives but also complicate the detection of legitimate gestures.

An additional method for avoiding false positive gesture events is to define gestures that require the user to move his hand over the sensor node multiple times as this reduces the likelihood of such light variations occurring randomly. Our implementation supports specifying dynamically how many times the user should move his hand over the sensor node as part of a gesture.

B. Interaction Protocol

The gesture detection on the wireless sensor nodes is always initiated by a trigger message, the *gesture announcement message*, which is broadcasted by the client node. One the one hand, this helps reducing the load on the sensor nodes as continuous sampling would be very expensive in terms of energy and processor cycle consumption. On the other hand, it greatly reduces the potential for false positives in the gesture detection as light level variations can only affect gesture-based interaction if they occur right after a client node has sent a gesture announcement message.

The gesture announcement message is received by all sensor nodes lying within the communication range of the client node. The nodes react by recording their current light level as their base sensor value l_{Base} before the actual gesture starts. Each node then continues to sample its light sensor every $t_{SampleInterval}$ time units and checks whether the expected sequence of "sensor covered" and "sensor uncovered" states can be detected within the specified time intervals. If this is the case, then it reports back to the client node with a *gesture detected message*. Otherwise, the sensor node times out and does not participate in the interaction with the client node.

V. Sensor Node Lamp Interaction

One drawback of gesture-based interaction is that it requires direct access to the sensor node to perform a gesture. With the Sensor Node Lamp interaction, we present an approach that overcomes this limitation with the help of a special hardware device, the Sensor Node Lamp (SNL).

A. Approach

Like gesture-based interaction, the SNL interaction mechanism uses the light sensor of a node to detect whether it is the target of an interaction. The idea is to have the user point the Sensor Node Lamp in direction of the destination node and use an integrated light source (a strong LED) to generate a pre-defined pattern of "light on" and "light off" events. The destination node is then able to detect the light pattern whereas the light levels of all other nodes not lying within the light beam of the SNL remain unaffected.

We have built a prototype of the Sensor Node Lamp that can be mounted on top of a TelosB sensor node (see Fig. 2). It consists of a powerful 1 watt LED mounted behind a focusing lens (8.7 degrees cone angle), a button for user input to the application, a separate power supply in the form of two AA batteries and a constant current transformer used to provide a constant light level over the lifetime of the batteries. The LED emits red light at a dominant wavelength of 625 nanometers

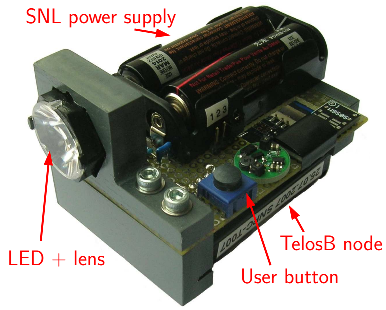

Fig. 2. Sensor Node Lamp

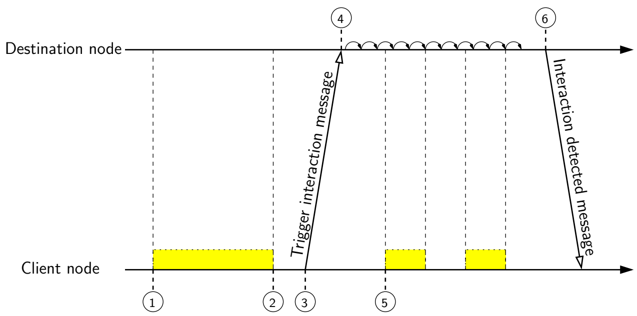

Fig. 3. Time diagram of an interaction using the Sensor Node Lamp

which both lies within the optimal reception range of the light sensors on the TelosB sensor nodes and is clearly visible to the user even in scenarios with a high ambient light level.

The SNL not only looks a little bit like a flashlight, it is also used in a very similar manner: Pressing the user button turns on the light. The light beam can then be directed at the object of interest by orienting the SNL accordingly. This **flashlight analogy** is an important aspect for the usability of the SNL as a flashlight is a well-known device most users are familiar with. Being able to explain the functionality of the SNL starting from the flashlight analogy greatly simplifies the explanation of the device and its working principle to new users. Another important advantage of the flashlight analogy is the immediate visual feedback that the SNL provides to the user. The user is able to see where he is pointing and which node he is targeting with the SNL.

The task that the user has to perform is kept intentionally simple. He only needs to point the device in the direction of the destination node and illuminate it with the SNL for a short time period. Everything else happens automatically in the background without requiring the user's attention.

B. *Interaction Protocol*

Fig. 3 shows the time diagram of a typical sequence of events for the interaction between client and sensor nodes using the Sensor Node Lamp. The user starts the process by pressing the user button on the SNL which activates the LED (1) to help the user in pointing the light beam in the direction of the destination node. Releasing the button deactivates the LED again (2) and starts the actual interaction protocol.

As a first step, the client node broadcasts a *trigger interaction message* (3) to alert neighboring nodes of the upcoming interaction. It then waits for a short time interval before sending a simple on-off-on-off light pattern (5) using its LED.

The sensor nodes react to the reception of a *trigger interaction message* by recording their base sensor value l_{Base} (4) which they use – together with a threshold distance δ_{Thresh} – to differentiate between "light on" and "light off" sensor readings. The nodes then continue to periodically sample their light sensors to detect the pattern sent by the SNL.

The destination node successfully detects the light pattern and informs the client by sending back an *interaction detected message* (6). All other nodes time out after $t_{TriggerMax}$, stop sampling their light sensors and do not participate in the rest of the interaction.

An additional challenge revealed in preliminary experiments lies in strong oscillations of the ambient light level in scenarios using fluorescent tubes for the illumination of rooms. Their light intensity oscillates with a frequency of 100 Hz (or 120 Hz) which – under seemingly stable conditions – causes the values recorded by the light sensors to vary within a certain interval $[a, b]$. If the base sensor value l_{Base} lies near a, then already the oscillation of values caused by the fluorescent tubes can trigger "light on" events for small values of δ_{Thresh} which in in turn can disrupt the light signal detection and the interaction setup with the SNL fails.

To deal with this problem, we have extended the procedure for recording l_{Base}: Instead of sampling only once, we sample five times in a row (with 2 ms wait time in between) and use the maximum of the five values as l_{Base}. With a high probability, this gives us a base value which lies near b and – as the evaluation shows – effectively eliminates the problem of false positive "light on" events during the signal detection.

A similar problem also exists for gesture-based interaction. There we use the same basic solution but take the minimum instead of the maximum of the five samples as the base value.

VI. NextOnePlease Interaction

The first two interaction approaches both rely on using light sensors to facilitate interaction. However, in some scenarios such light sensors might not be available or using them might interfere with the normal tasks of the sensor nodes. In this section, we discuss *NextOnePlease* (NOP), an interaction solution that works without using any sensor information.

A. *Approach*

The idea of *NextOnePlease* is to use the small, controllable LEDs available on most sensor node platforms to provide visual hints to the user that allow him to select the correct node he wants to interact with.

NextOnePlease allows the user to browse through a list of the nodes in his neighborhood (sorted based on the context of the client) until he finds the destination node. *NextOnePlease* highlights one node at a time by turning its LEDs on. The user can then either select this node or ask for the next one – both by simply pressing a button. Selecting a specific node

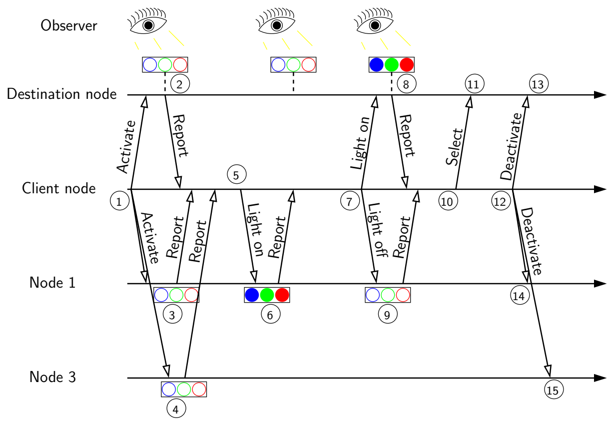

Fig. 4. Time diagram of an example NextOnePlease interaction

therefore only requires to repeatedly press a button until the LEDs on this node are turned on – a simple task that can be performed very quickly.

B. Interaction Protocol

Fig. 4 illustrates the *NextOnePlease* protocol with a typical sequence of events in a time diagram. The client node initiates the procedure by broadcasting an *activate message* (1). This triggers all sensor nodes within the communication range of the client node to start sending back periodic *report messages* (2-4). These *report messages* allow the client node to determine the set of candidate nodes which it sorts in a list.

Now, each time the user presses the "Next node" button, the client node sends a *light on message* to the next node in its list while sending a *light off message* to the previously highlighted node (5,7). The sensor nodes react accordingly to these messages by turning their LEDs either on or off (6,8,9). This process continues until the user detects that the LEDs of the destination node are turned on and presses the "Select current node" button. On the protocol level, this triggers a *select message* being sent to the destination node (10-11). The user is then able to either select additional nodes or to finish the selection process. For doing this, the client node broadcasts a *deactivate message* (12) whose reception (13-15) stops the periodic sending of report messages.

C. Implementation

We have implemented *NextOnePlease* on TelosB sensor nodes. They provide a red, a green and a blue LED which we all turn on or off at the same time. The client application runs on Sharp Zaurus PDAs and can be controlled either through a graphical user interface or with the help of two hardware buttons on the side of the PDA.

To improve the behavior of *NextOnePlease* in situations with very large numbers of candidate nodes, the client application sorts the internal node list based on the received signal strength (RSSI) values of the report messages. The underlying intuition is that the user likely wants to interact with nodes in his direct vicinity whose report messages should be received with a higher RSSI value than for nodes farther away.

TABLE I
SIZE OVERHEAD ANALYSIS (ALL VALUES IN BYTES)

		Blink	Oscill. Temp	Oscill. Light	MViz
Original	ROM	2656	13442	16608	28150
	RAM	55	394	438	1912
Gesture	ROM	17498	19662	18674	29914
	RAM	528	634	602	2074
SNL	ROM	17298	19416	18442	29652
	RAM	510	616	584	2060
NOP	ROM	12084	14222	17370	28734
	RAM	354	462	506	1980

As the RSSI value of messages is subject to many environmental influences that distort the distance/signal-strength relationship [16], we only use the RSSI value to partition the nodes into three groups: "Nearby", "medium distance", "far away". Nodes are assigned to the groups based on a moving average of their RSSI values recorded over time and the nodes are then browsed group by group.

VII. EVALUATION

In this section, we show and discuss results from evaluating our three sensor node interaction approaches.

A. Memory Overhead

The memory consumption of our interaction mechanisms is a relevant factor as both program memory and main memory are very limited resources on typical sensor node platforms (e.g., 48 kB of program memory and 8 kB of main memory on the TelosB nodes).

The actual overhead depends on the application the mechanisms are integrated with as the interaction mechanism can share code with other application parts running on the sensor node (e.g., radio communication or sensor access modules). To evaluate this, we have integrated gesture-based interaction, SNL interaction and NOP interaction with three applications from the TinyOS source tree: **Blink**, **Oscilloscope** (one configuration using the temperature sensor; one using the light sensor) and **MViz** (using the light sensor). Table I summarizes the resulting size values for both program memory (ROM) and the main memory (RAM).

The more of the required modules the application already contains, the smaller the overhead of integrating the interaction mechanisms; ranging from the small Blink application which only includes minimal functionality by itself (i.e., no sensor access, no communication) to the complex MViz application which allows reusing large parts of the code. Overall, the memory overhead of all three approaches is reasonably small (e.g., only between 584 bytes and 1764 bytes of program memory and between 68 bytes and 162 bytes of main memory when integrated with MViz) and should allow the integration with a variety of applications.

NextOnePlease has the lowest memory requirements among the three interaction mechanisms. This was expected as it does not require accessing sensor information. Moreover, the implementation on the sensor node can be kept simple as the actual application logic for the sensor node selection is implemented on the client device.

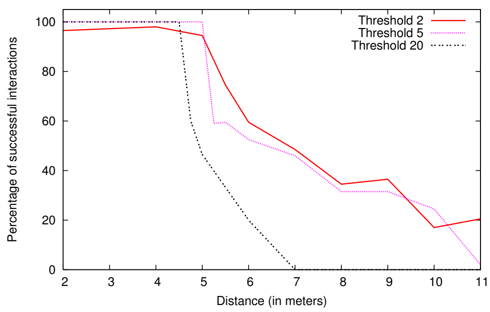

Fig. 5. SNL success rate over distance

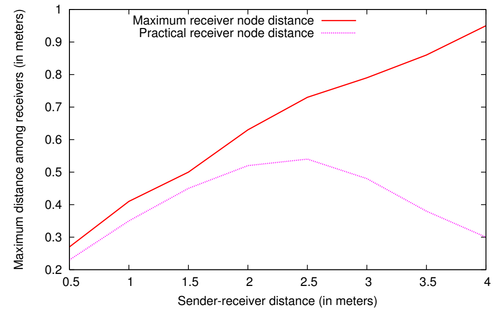

Fig. 6. Maximum distance among SNL receiver nodes

B. Interaction distance

The maximum distance between the user and the node he wants to interact with is an essential property as it determines in which situations the interaction mechanism can be used.

1) Gesture-based Interaction: Gesture-based interaction has the strongest limitations with respect to the interaction distance as the user needs to directly approach the node for performing the gesture. Our experiments showed that a distance of 5 centimeters between the hand and the sensor node is reasonable. While larger distances could be supported by decreasing δ_{Thresh}, this would greatly increase the potential of false positive gestures being detected.

2) SNL Interaction: For the SNL, we evaluated the maximum interaction distance and the success rate of interactions at different distances in a controlled experiment placing SNL and receiver node at specific distances of each other in the normally lit hallway of a building. We then measured the success rate of SNL interactions based on 200 trials per distance. Fig. 5 shows the results for distances between 2 and 11 meters for three different values of δ_{Thresh}. Note that we are using raw sensor and threshold values here instead of meaningful light units to avoid any conversion overhead on the nodes.

When the distance between SNL and receiver node increases, the impact of the SNL on the light sensor decreases. For $\delta_{Thresh} = 5$ and $\delta_{Thresh} = 20$, we were able to achieve a 100% success rate for distances up to 5 meters and 4.5 meters respectively. Above these threshold distances, our experiments showed a sharp decline of the success rate. However, we were still able to initiate some interactions up to a distance of 6 meters ($\delta_{Thresh} = 20$) or 11 meters ($\delta_{Thresh} = 5$). This behavior can be explained as follows: Above a certain distance, the difference δ_{Impact} between the light levels in the “light on” and the “light off” states becomes too small to compensate the continuous oscillations of the light levels in all cases. Whether a “light on” event can be detected now depends on the timing of the background oscillations while the light values are sampled. The smaller δ_{Impact} becomes with an increasing distance, the more situations exist where the measured light level does not suffice for a “light on” event.

Fig. 5 also shows an example for δ_{Thresh} being selected too small ($\delta_{Thresh} = 2$) which allows normal light level variations to trigger light events. As we are checking the timings of the SNL light signal on the receiver side, false positives are still effectively prevented and we could not detect any during extensive experiments. However, untimely light events can disturb the detection of an actual SNL light signal and impede a successful interaction setup. This is illustrated in Fig. 5 for a threshold of 2: Even for small distances, a 100% success rate cannot be achieved as external light events interfere with some of the signal detections. For larger distances, however, the smaller threshold is able to produce a slightly higher success rate than for a threshold of 5.

With a maximum interaction distance of around 5 meters, SNL interaction can be used comfortably in typical indoor scenarios even within quite large rooms.

Besides the maximum distance between the SNL and the receiving sensor node, the minimum distance among nodes that are to be addressed individually from a given distance is also an important usability factor. We determined this resolution of our selection method by measuring at which distance among two receivers we were still able to trigger an interaction simultaneously on both nodes. Fig. 6 shows the results plotted over the sender-receiver distance. As shown in the figure, the maximum receiver node distance stays below one meter for a sender-receiver distance of four meters. This confirms that the SNL allows for an effective interaction with individual nodes even for relatively dense deployments.

If the user user wants to select two nodes at the same time, then the maximum receiver distance is less relevant than the distance at which such a simultaneous selection works reliably. We defined this practical receiver node distance as the maximum distance among receiver nodes at which we were able to perform at least five successful interactions in a row. The results are also plotted in Fig. 6. The practical receiver node distance only grows up to a little more than 0.5 meters (at a sender-receiver distance of 2.5 meters) and then falls again as the decreasing light level influence of the SNL is only sufficient for reliably triggering interactions near the center of the light beam.

3) NextOnePlease Interaction: For *NextOnePlease*, the maximum interaction distance is the maximum distance from which the user is still able to determine whether the LEDs on the destination node are turned on or off. This clearly depends on the properties of the LEDs used on the node, the specific scenario (i.e., the lighting conditions) and also on the user himself. To get an idea of the distances possible, we set up one indoor experiment in a hallway and one experiment outdoors using TelosB sensor nodes. We measured the maximum distance at which it was still possible to tell whether the LEDs of a node were turned off or on.

For the indoor scenario, we were still able to tell the LED states apart from a distance of more than 50 meters! As expected, the maximum distance in the outdoor scenario depended on the specific settings. In the worst case, bright sunlight shining directly at the node, detection of the LED states was possible up to a distance of three meters. With the sun coming from the side or from behind the node, the maximum interaction distance lay between 8 and 15 meters.

Overall, the experiments have shown that distance is not a limiting factor for *NextOnePlease* indoors and that interactions are also possible using this method in outdoor scenarios.

C. User Study

To learn more about how good our approaches support users in interacting with wireless sensor nodes, we performed a small user study in which we asked 12 participants recruited from students and staff to use each of the three systems to execute a set of interaction tasks. Before the experiment started, each participant got an explanation and a short demonstration of how the three interaction mechanisms work.

As the basic experimental setup, we placed 12 TelosB sensor nodes on the wall of a large room arranged in a 6x2 Cartesian grid with an edge length of one meter. For each run, we placed colored signs at three randomly selected nodes to mark them as destination nodes and asked the participant to select these nodes for interaction using gesture-based interaction, Sensor Node Lamp interaction and *NextOnePlease*. If an interaction attempt failed, we asked the user to try it again until being successful. The participants repeated each experiment three times with different nodes being used as destination nodes.

During the experiments, we were mainly interested in investigating two aspects: First, we measured the **required time** for completing each interaction task for assessing the overhead of each method. Second, we looked at the **success rate** of performing the interaction.

1) Interaction Times: Fig. 7 summarizes the results of the times measured in our experiments showing both the average time required by the participants as well as the maximum and minimum times. Interacting with the help of the SNL clearly outperforms both gesture-based interaction and *NextOnePlease*. With only 6.82 seconds required on average for selecting three nodes, the participants completed the SNL interactions nearly two times faster than with *NextOnePlease* and 2.8 times faster than with gesture-based interaction. There also do not exist significant outliers for SNL whereas the longest run with *NextOnePlease* took more than 30 seconds and more than 45 seconds for gesture-based interaction.

Considering the complexity of the interaction tasks (selecting three nodes in a row), the average interaction times of all three approaches fulfill our expectations and indicate that an efficient use is possible in real-world scenarios.

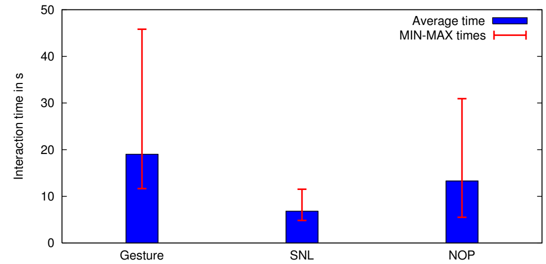

Fig. 7. Interaction times

2) Success Rate: Failures of gesture-based interactions were mainly due to users moving their hand too quickly over the nodes or keeping too much distance to the light sensors of the nodes. However, some failures were also caused by gesture announcement messages being lost. Here, the reliability of message reception was negatively influenced by the often very small distance between the client device and the receiving node when sending the message. Overall, 89.4% of the interaction attempts succeeded. There was also a small number of false positives (in 4.6% of the cases) due to the small distance among the nodes in the experiment. They were mainly caused by the shadow of users interacting with neighboring nodes.

Only very few interaction attempts with the SNL failed resulting in a success rate of 97.22%. They were all easily and quickly recovered in a second attempt. Only one participant managed to create a false positive event once activating two neighboring nodes at the same time (0.93% false positives).

False negatives do not occur for interactions with *NextOnePlease* as any destination node the user misses while clicking through the list of nodes will be offered again in the next round only resulting in a longer interaction time. As also no participant falsely selected a node, we achieved a 100% success rate for *NextOnePlease*.

Since repeating failed interaction attempts is simple and inexpensive, the success rate of all three approaches qualify them for practical use. For gesture-based interaction, we also expect a higher success rate achievable by experienced users.

3) User Survey: After completing the experiment with all three systems, we let the participants fill out a short questionnaire asking about their experience with gesture-based interaction, SNL interaction and *NextOnePlease*. They rated the initial difficulty of understanding the working principle of the mechanisms (from -2 (“Very difficult”) to +2 (“Very simple”)) and the convenience of using the interaction mechanisms (from -2 (“Very inconvenient”) to +2 (“Very convenient”)).

Our participants were very satisfied with both the ease of understanding and the convenience of use of the SNL interaction (1.83 average rating in both categories). *NextOnePlease* was also easily understood (1.42 points) by the users but was considered more critically in terms of convenience (-0.08 points). Finally, our participants rated gesture-based interaction near the median of the scale both for the ease of understanding (0.5 points) and the convenience of use (0.0 points).

We attribute some of the dissatisfaction of the users with

TABLE II
SELECTION CRITERIA

	Gesture	SNL	NOP
Hardware required	Light sensors	Light sensors + SNL	LEDs
Interaction distance	$\leq 5cm$	$\leq 5m$	$\leq 8m$
User effort	Medium	Low	Medium
Scenarios	Indoor+Outdoor	Mainly indoor	Indoor+Outdoor
Scalability	Good	Good	Limited

gesture-based interaction and *NextOnePlease* to the prototype status of the client device used. The Zaurus PDA with an attached TelosB sensor node (required as a communication bridge to the sensor network) is not as handy as the self-contained SNL device.

For *NextOnePlease*, a special difficulty of the interaction task lay in selecting three nodes. The participants tried to observe all three nodes in parallel which made the simple task of browsing through the list of nodes much more challenging and exhausting than when concentrating on a single node. However, we expect that most real-world tasks will involve the selection of a single node at a time.

We also learned that gesture-based interaction poses a more difficult initial learning experience than originally expected. While we as expert users were very comfortable and efficient in using gesture-based interaction, the participants of our study struggled with the different errors a user can make in performing gestures (e.g., moving too fast).

D. Comparison of Approaches

Gesture-based interaction, Sensor Node Lamp interaction and *NextOnePlease* focus on different aspects and work best in different scenarios. Table II summarizes relevant criteria for the selection of one of the interaction mechanisms.

Based on our experience and the results of our user study, **SNL interaction** is the preferable method for interacting with wireless sensor nodes. It works reliably, is simple to use, scales well with the size of the network and is able to cover reasonable distances. Its main limitations are that it works mainly indoors and that it requires additional hardware on the client side. The main strengths of **gesture-based interaction** are its low hardware requirements and its ability to work both indoors and outdoors. However, the small maximum interaction distance limits the set of scenarios it is usable in. Finally, ***NextOnePlease*** also sets low hardware requirements (LEDs), works at reasonable distances and can be used indoors and outdoors. Besides the required effort by the user, its main challenge is the scalability in the presence of a very large number of nodes.

VIII. CONCLUSIONS

In this paper we have discussed the problem of initiating interactions between mobile devices and wireless sensor nodes in pervasive computing scenarios. We have introduced three different solution approaches: a gesture-based approach, a solution using light signals sent from the client device to the sensor nodes and an approach that is based on sensor nodes providing information using their LEDs. We have shown that these solutions cover a wide range of scenarios building on different node capabilities and user actions. In the evaluation we have demonstrated that all three mechanisms are effective in solving the sensor node interaction problem. Sensor Node Lamp interaction, in particular, has been shown to be a very efficient and easy-to-use solution for indoor scenarios.

As part of future work, we would like to investigate the applicability of our solutions to other devices that are part of pervasive computing scenarios like for example small consumer appliances. For *NextOnePlease*, we are interested in improving the scalability and the usability with more advanced ways of sorting the list of nodes presented to the user, for example based on the role of nodes or based on the history of previous queries of the user. For the Sensor Node Lamp, we are working on more ways of using the SNL as an actuator in various sensor network scenarios.

REFERENCES

[1] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison, “Bridging physical and virtual worlds with electronic tags,” in *Proc. of the SIGCHI Conf. on Human factors in Computing Systems*, 1999.

[2] E. Rukzio, K. Leichtenstern, V. Callaghan, P. Holleis, A. Schmidt, and J. S.-Y. Chin, “An experimental comparison of physical mobile interaction techniques: Touching, pointing and scanning,” in *Proc. of the 8th Int. Conf. on Ubiquitous Computing*, 2006.

[3] K. Mäkelä, S. Belt, D. Greenblatt, and J. Häkkilä, “Mobile interaction with visual and rfid tags: a field study on user perceptions,” in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, 2007.

[4] T. Starner, J. Auxier, D. Ashbrook, and M. Gandy, “The gesture pendant: A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring,” in *Proc. of the 4th Int. Symp. on Wearable Computers*, 2000.

[5] C. Metzger, M. Anderson, and T. Starner, “Freedigiter: A contact-free device for gesture control.” in *Proc. of the 8th Int Symp. on Wearable Computers*, 2004.

[6] J. Rekimoto, “Gesturewrist and gesturepad: Unobtrusive wearable interaction devices,” in *Proc. of the 5th Int. Symp. on Wearable Computers*, 2001.

[7] P. Föckler, T. Zeidler, B. Brombach, E. Bruns, and O. Bimber, “Phoneguide: museum guidance supported by on-device object recognition on mobile phones,” in *Proc. of the 4th Int. Conf. on Mobile and Ubiquitous Multimedia*, 2005.

[8] D. Merrill and P. Maes, “Augmenting looking, pointing and reaching gestures to enhance the searching and browsing of physical objects.” in *Proc. of the 5th Int. Conf. on Pervasive Computing*, 2007.

[9] A. Ghali, S. Benford, S. Bayoumi, J. Green, and T. P. Pridmore, “Visually-tracked flashlights as interaction devices,” in *Proc. of the IFIP TC13 Int. Conf. on Human-Computer Interaction (INTERACT 03)*, 2003.

[10] H. Ma and J. A. Paradiso, “The FindIT Flashlight: Responsive tagging based on optically triggered microprocessor wakeup,” in *Proc. of the 4th Int. Conf. on Ubiquitous Computing*, 2002.

[11] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen, “Smart-its friends: A technique for users to easily establish connections between smart artefacts,” in *UbiComp ’01: Proc. of the 3rd Int. Conf. on Ubiquitous Computing*, 2001.

[12] S. N. Patel, J. S. Pierce, and G. D. Abowd, “A gesture-based authentication scheme for untrusted public terminals,” in *UIST ’04: Proc. of the 17th Symp. on User Interface Software and Technology*, 2004.

[13] R. Peterson and D. Rus, “Interacting with sensor networks,” in *Proc. of the IEEE 2004 Int. Conf. on Robotics and Automation*, 2004.

[14] J. Lifton, M. Mittal, M. Lapinski, and J. A. Paradiso, “Tricorder: A mobile sensor network browser,” in *Proceedings of the ACM CHI 2007 Conference - Mobile Spatial Interaction Workshop*, April 2007.

[15] O. Saukh, R. Sauter, J. Meyer, and P. J. Marrón, “Motefinder: A deployment tool for sensor networks,” in *Proc. of the ACM Workshop on Real-World Wireless Sensor Networks (REALWSN’08)*, 2008.

[16] K. Whitehouse, C. Karlof, and D. Culler, “A practical evaluation of radio signal strength for ranging-based localization,” *SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 1, pp. 41–52, 2007.