ESF Exchange Grant – Final Report

Efficient Signal Strength Sampling for 802.11-based Indoor Positioning Systems

Thomas King, University of Mannheim, Germany

1 Purpose of the Visit

The purpose of my stay at the University of Aarhus was twofold:

• *Initiate a Cooperation:* One goal of my past stay was to initiate a cooperation between the University of Aarhus and the University of Mannheim. On both side, research groups work on location-aware middleware to facilitate the development of location-based services for mobile devices. Because of our complement knowledge my stay was beneficial for both research groups.

• *Progress in Research of 802.11-based Positioning Systems:* During my stay we wanted to answer and actually answered the following questions:

   – What is the impact of scanning on the position accuracy of 802.11-based positioning systems?
   – How does regular scanning interfere with interactive network applications?
   – Is the running time of a mobile device significantly impacted by regular scanning?
   – Can we find an algorithm to adaptively select the scanning interval based on movement of the mobile device?
2 Description of the Work Carried Out During the Visit

The work load carried out during my stay can be categorized in five phases. Each phase is approximately two weeks in length and partly overlapping with the preceding and successive phase. The phases are as following:

- **Data Collection:** Mr. Kjærgaard and I repetitively collected a large number of signal strength measurements with different network cards, with different scanning technologies, at different locations, and during different times of the day.
- **Feature Selection and Emulation:** We implemented feature selection algorithms as described in the literature. Further, we implemented our own ideas. Based on the data collected in the previous phase we run emulations to evaluate the algorithms.
- **Prototype Implementation:** The best feature selection algorithms are implemented as a prototype.
- **Real-World Validation:** The prototype implementation is utilized to validate the results obtained by emulation in real-world deployments.
- **Paper Writing:** Finally, the results are summarized and put down to the paper entitled “ComPoScan: Adaptive Scanning for Efficient Concurrent Communications and Positioning with 802.11”.

3 Description of the Main Results Obtained

The main result of our cooperation is the novel ComPoScan system that can mitigate the effect of scanning on concurrent communications. ComPoScan is based on movement detection to adaptively switch between lightweight monitor sniffing and invasive active scanning. Additionally, during my stay we evaluated the system both by emulation and by validation in a real-world deployment. The emulation showed that our movement detection system works independently of the environment, the network card, the signal strength measurement technology, and number and placement of access points. During our experiments, we also showed that ComPoScan does not harm the positioning accuracy of the positioning system. By validation in a real-world deployment we provided evidence for that the real system works as predicted by the emulation. In addition, we provide results for
ComPoScan’s impact on communication where it increased throughput by a factor of 122, decreased the delay by a factor of ten, and decreases the percentage of dropped packages by 73%.

4 Future Collaboration with Host Institution

During my stay at the University of Aarhus, Mr. Kjærgaard and I discussed the need of research in the area of dilution of precision metrics for 802.11-based positioning systems. These metrics should inform a user about the current imprecision of the system. While we started talking about this topic a few weeks ago we currently discuss the working package for our joint-efforts. However, so far, we are still in the planning phase but we hope we are able to start with the research work soon.

5 Projected Publications

Attached in the appendix please find the paper entitled “ComPoScan: Adaptive Scanning for Efficient Concurrent Communications and Positioning with 802.11”. This paper has been submitted to the Sixth ACM International Conference on Mobile Systems, Applications, and Services (ACM MobiSys). The notification of acceptance is scheduled for 4th March 2008.

6 Appendix

- ACM MobiSys 2008 Paper Submission:
ComPoScan: Adaptive Scanning for Efficient Concurrent Communications and Positioning with 802.11

Thomas King
Department of Computer Science
University of Mannheim, Germany
king@informatik.uni-mannheim.de

Mikkel Baun Kjærgaard
Department of Computer Science
University of Aarhus, Denmark
mikkelbk@daimi.au.dk

ABSTRACT

Using 802.11 concurrently for communications and positioning is problematic, especially for real-time communications (e.g., VoIP, video conferencing). Periodical scanning for measuring the signal strength interrupts the data flow. Reducing the scan frequency is also no option because it hurts the position accuracy. For this reason, we need an adaptive technique to mitigate this problem.

This work proposes ComPoScan which based on movement detection adaptively switches between light-weight monitor sniffing and invasive active scanning to allow positioning and minimize the impact on the data flow. The system is configurable to realize different trade-offs between position accuracy and the level of communication interruption.

We provide extensive experimental results by both emulation on data collected at several sites and validation in several real-world deployments. Results from the emulation show that the system can realize different trade-offs by changing parameters. Furthermore, the emulation shows that the system works independently of the environment, the network card, the signal strength measurement technology, and number and placement of access points. We also show that ComPoScan does not harm the positioning accuracy of a positioning system. By validation in several real-world deployments we provided evidence for that the real system works as predicted by the emulation. In addition we provide results for ComPoScan’s impact on communication where it increases throughput by a factor of 122, decreases the delay by a factor of ten, and decreases the percentage of dropped packages by 73%.

1. INTRODUCTION

Back in 1999, when IEEE 802.11 was being standardized, the researchers and engineers working on the standard probably never thought about the new ways we use this technology today. Real-time applications such as voice over IP and video conferencing were years ago a rarity but are nowadays a common day to day phenomenon. These real-time applications have hard requirements in terms of bandwidth, delay, and packet loss to be functional. An even more extreme new way of usage is to utilize the signal strength measurement capabilities of 802.11 network cards as basis for indoor positioning systems. Initially, signal strength measurements are performed during a so-called active scan to let a network card decide which access point might be the best to connect to. Many indoor positioning systems (e.g., [2, 6]) make use of 802.11 because almost all modern cell-phones and laptops are equipped with this wireless technology. Therefore, the devices can be used for positioning as they come out of the box which means no additional hardware is required.

Even the newer sub-standard 802.11b and 802.11g do not satisfy all these requirements. Furthermore, many workarounds and novel approaches (e.g., [5, 13, 14]) have been proposed to make 802.11 ready for many of these new requirements. However, still unsolved is the problem that occurs when a 802.11 network card is used for positioning and communications at the same time. On one side the positioning system requires as steady stream of active scans to be able to deliver accurate position estimates. Performing an active scan means that the network card switches through all the different channels in search for access points. Depending on the network card this takes around 600 milliseconds. During this time no communication is feasible. On the other side are the demanding real-time applications. For instance, a video conference requires depending on the video and voice quality around 512 KBit/s of bandwidth and a round trip delay of less than 200 milliseconds [17].

Figure 1 depicts what happens to throughput and delay of a 802.11g-enabled mobile device if the network card is requested to perform an active scan every 600 milliseconds. During the first 20 seconds communication is untroubled which means on average a throughput of 15.77 MBit/s and a round trip delay of less than 42.03 millisecond is achievable. In the 20th second active scanning kicks in. The remaining seconds provide only 0.1 MBit/s of throughput and 532.11 milliseconds of delay because active scans are performed so often.

Figure 1: Throughput and delay.

In this paper we propose a novel solution called ComPoScan to this problem. ComPoScan is based on movement detection to adaptively switch between light-weight monitor sniffing and invasive active scanning. Only, if the system detects movement of the user active scans are performed to provide the positioning system with the signal strength measurements it needs. If the system detects that the user is standing still it switches to monitor sniffing to allow communications to be uninterrupted.
Monitor sniffing is a novel scanning technique proposed in [8]. It works with most 802.11 network cards used today. Monitor sniffing allows a mobile device to recognize access points operating on channels close to the one it is using for communications with the access point it is associated with. It has been shown that up to seven channels can be overheard without any disturbance of the actual communication.

Our movement detection approach is also based on signal strength measurements. However, the measurements provided by monitor sniffing are sufficient to reliably detect if the user is moving or standing still. We designed the movement detection system to be configurable so that depending on the user’s preferences communication capabilities or positioning accuracy can be favored.

We make the following contributions in this work: First, we are the first to present a system to mitigate the effect of scanning on concurrent communications. Second, we are the first utilizing monitor sniffing and active scanning to build a reliable indoor movement detection system. Third, we provide a deep investigation by means of emulation to show that our movement detection system works independently of the environment, the network card, the signal strength measurement technology, and number and placement of access points. Additionally, we show that it does not harm the positioning accuracy of the positioning system. Fourth, we implement ComPoScan and use this prototype in a real-world deployment to gather results showing that the real system works as predicted by the emulation. Our results show that our goal of mitigating the effect of scanning on communications is fulfilled.

The remainder of this paper is structured as follows: In Section 2 we present the related work. Subsequently, we introduce our novel ComPoScan system. The details of our movement detection approach are discussed and evaluated by means of emulation in Section 4. Section 5 discusses our prototype implementation of ComPoScan in detail. The results of our real-world deployment are presented in Section 6. Finally, Section 7 provides a discussion and Section 8 concludes the paper and provides directions for future work.

2. RELATED WORK

As mentioned earlier existing 802.11 positioning systems (e.g., [2, 6]) have not considered the problem of concurrent communication and positioning. As a central part of the ComPoScan system we apply movement detection to address this problem.

The first and as far as we know the only 802.11-based system that emphatically focuses on movement detection is the LOCADIO system [11]. In their paper, the authors propose an algorithm that exploits the fact that the variance of signal strength measurements increases if the mobile device is moved compared to if it is still. To smooth the high frequency of state transitions a HMM is applied. The results in the paper show that the system accurately detects whether or not the mobile device is in motion in 87 percent of all cases. Compared to our approach the authors do not compare their approach to other movement detection algorithms. Furthermore, the results are only based on emulation which means that the signal strength data is collected in a first step and then later analyzed and processed to detect movement. This is a valid approach but it might happen that some real-world effects are missed. Another fact that the authors of the aforementioned paper do not look into is the impact of periodic scanning to the communication capabilities of mobile devices at all. They just assume that a 802.11 network card is solely used for movement detection. Finally, all results are based on one single client. Which means that variations in signal strength measurements caused by different wireless network cards are not taken into account.

Two GSM-based systems have also been proposed by Sohn et al. [16] and Anderson et al. [1]. The system by Sohn et al. is based on several features including variation in Euclidean distance, signal strength variance and correlation in strength ranking of cell towers. The system classifies data into the three states of still, walking and driving. By emulation on collected data they achieve an overall accuracy of 85%. The system by Anderson et al. detect the same states but uses the features of signal strength fluctuation and number of neighbouring cells. Using these features they achieve a comparable overall accuracy to the former system. As for LOCADIO the results for both systems are only based on emulation, they also do not consider communication and the results are based on one client.

3. COMPOSSCAN SYSTEM

For our system we assume that the mobile device that should be ComPoScan-enabled contains a 802.11 network card. This card should be able to perform active scans and monitor sniffs on a high rate (e.g., every 600 milliseconds). Further, the card should not include buffered results from a previous scan into the current scan result. For the area where ComPoScan should be deployed we assume that at least one access point is recognizable by monitor sniffing and active scanning at all times.

Our main goal for ComPoScan is to minimize the impact of scanning on concurrent communications. For this, we want to build a movement detection system that based on signal strength measurements provided by monitor sniffing or active scanning detects correctly whether the user is standing still or moving. If this would be possible only active scans are required if the user is roaming around. But we expect that it might be impossible to built a completely perfect movement detection system with 802.11. So, this brings up a sub-goal: The movement detection system should be configurable in such a way that the user can define the kind of the error the movement detection system is producing. If the user is more interested in precise position estimates than in uninterrupted communications then this scenario should be configurable. Also, the other way around should be supported.

ComPoScan works as illustrated in Figure 2. At startup active scans are performed to collect signal strength values from as many access points as possible. Based on this data the current state is calculated. If the system detects movement it performs another active scan. In case the system draws the conclusion that the user is standing still it switches to monitor sniffing for signal strength measurement. Based on this data the current state is reevaluated and the system starts over again.

4. MOBILITY DETECTION

A central part of the ComPoScan system is movement detection based on signal strength. This section describes our experimental setup, gives an analysis of features used for movement detection, presents the used method and discusses our emulation results.
4.1 Experimental Setup

For our experimental setup we describe the used hardware and software setup, the test environments and the details of the data collection process.

4.1.1 Hardware and Software Setup

To collect the signal strength measurements we used an IBM Thinkpad R51 laptop running Linux kernel 2.6.22.12 and Wireless Tools 29pre22. To show that our approach works independent of a particular card we use different network cards. For this, three network cards were chosen that are all quite commonly used today. We selected a Lucent Orinoco Silver PCMCIA card, a TRENDnet TE-W-501PC PCMCIA card, and an Intel Centrino 2200 mini-pci card. The Lucent Orinoco card is a 802.11b only card. The TRENDnet card is based on the widely used Atheros AR5006XS chip-set and supports 802.11b, 802.11g, and 802.11a. Only 802.11b and 802.11g are supported by the Intel Centrino chip. However, all three network cards can be used for our purposes because they all support monitor sniffing and active scanning.

For the Intel Centrino 2200 card we used the ipw2200 driver in version 1.2.0. In the default settings the driver caches a scan result for 3.45 seconds which means that an access point that has been seen during the last 3.45 seconds will appear in a subsequent scan result even that it might be out of communication range. Because this property harms our movement detection system we modified the driver to discard old scan results before a new scan is performed.

The driver of the TRENDnet card needed also modifications. For this card we used the madwifi driver version 0.9.3.3. In the default settings the driver caches scan results in the same way as the ipw2200 driver. The difference here is that the cache timeout is even longer and set to 60 seconds. With our modifications the driver purges the cache before initiating a new scan. Because of the TWE-501PC card supports three 802.11 sub-standards it scans all the channels provided by 802.11b/g and 802.11a if a scan is initiated. As 802.11a access points are quite rare and not deployed at all at the environments where we collected signal strength measurements we wanted to stop the card from scanning 802.11a channels. For this, we restricted the driver to scan only 802.11b/g channels. During our analysis we realized that the driver scans only these channels actively which have been recently used by access points. The recently unused channels are only scanned passively. This behavior disturbs our approach because it might happen that access points that moved into communication range will not instantly be recognized. We solved this problem by forcing the driver to actively scan all channels. To improve the scanning speed we reduced the dwelling time the card is waiting for responses from access points at each channel to 10 milliseconds. The default settings randomly chose between 5 and 50 milliseconds. Furthermore, the driver cancels an ongoing scan as soon as application data emerges to be transmitted. During our bandwidth measurements the driver completely stopped scanning because data was always available to be delivered. To break this habit of the driver we completely disabled this features and modified the driver so that it performs a scan whenever it is asked to do so.

The orinoco_cs driver version 0.15 for the Lucent Orinoco card is unchanged because it behaved as required for our purposes.

The signal strength measurements are collected by using Loclib and Locana [7]. Loclib is a library that provides methods to invoke a scan and returns signal strength measurements collected from the driver of the selected network card. This data is then forwarded to the so-called Tracer application of the Locana software suite. Tracer visualizes signal strength measurements while they are taken. Furthermore, Traces stores the measurements together with user generated data such as position information into a file for further processing. We enhanced Tracer to update position information while scans are performed. This was required to be able to take measurements while roaming around.

4.1.2 Local Test Environments

We collected signal strength measurements in two different environments: On the second floor of the Hooper building and in a large hall at the ground floor of the Benjamin building at the University of Aarhus. The former environment is an newly build office building consisting of many offices (see Figure 3(a)). During a typical day many people move around. The area is covered by 22 access points of different vendors. We also deployed a 802.11-based positioning system on this environment covering an area of 55.7 times 12.7 meters. The blue dots in Figure 3(a) depict the positions where data for the fingerprint database has been collected.

The latter environment is an old warehouse building refitted to a lecture hall which means the place is scattered with tables and chairs (see Figure 3(b)). The hall is 26.3 meters in length and 15 meters in width. During our measurements only the people who collected the data where inside the room. The place is covered with 24 access points.

4.1.3 Data Collection

For the two test environments we collected signal strength data with two network cards at the same time. One network card uses monitor sniffing, the other active scanning. This allows us to directly compare signal strength measurements taken by monitor sniffing and active scanning because they are collected at the same time in exactly the same scenario. The network cards perform an active scan or a monitor sniff every 600 milliseconds.

To be able to compare different network cards we collected data for each environment with two different hardware configurations. The first configuration uses the Intel Centrino and the Lucent Orinoco network cards. The Centrino network card is performing monitor sniffing and the Orinoco card carries out active scans. For the second configuration the Centrino card is configured to perform active scans and the TRENDnet card collects data by using monitor sniffing.

---

1http://ipw2200.sf.net
2http://www.madwifi.org
3http://www.nongnu.org/orinoco/
The second floor of the Hopper building. The fingerprint database is marked in blue and the movement track is depicted in red.

Figure 3: Ground plans for the two local test environments.

For each test environment and each hardware configuration we collected four different movement scenarios. The four scenarios consist of two slow walking scenarios and two fast walking scenarios. Slow walking is defined as average movement speed below 0.7 m/s. Fast walking is defined if the movement speed is above this threshold. One of the slow walking and one of the fast walking scenarios comprise two movement transitions and the other scenarios include nine. A movement transition is defined as start walking or stop walking. The percentage of time where the person is still is varied between different scenarios. We selected the parameters for these scenarios in such a way that the parameter space containing movement speed, number of transitions, and percentage of time where the person is still is masked. All the scenarios are listed and described in Table 1. This table also names typical representative for this kind of movements. The representativeness were what we had in mind when defining the movement scenarios to collect data.

Table 1: Description of the different scenarios used for data collection.

<table>
<thead>
<tr>
<th>Speed</th>
<th>Transitions</th>
<th>% being still</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slow</td>
<td>2</td>
<td>90%</td>
<td>Meeting attendant</td>
</tr>
<tr>
<td>Slow</td>
<td>9</td>
<td>40%</td>
<td>Student working in a lab</td>
</tr>
<tr>
<td>Fast</td>
<td>9</td>
<td>40%</td>
<td>Student during lunch break</td>
</tr>
<tr>
<td>Fast</td>
<td>9</td>
<td>90%</td>
<td>Office worker</td>
</tr>
</tbody>
</table>

To be able to investigate the impact of different times of the day we collected all possible options once during typical office hours and once during evenings. In total we collected more than 11 hours of signal strength measurements.

We used the aforementioned Tracer application to trace the walks of the persons collecting data. For this, we stucked labels on the floor so that each corner and each dead end of the walking track is marked. Each time the person carrying out signal strength measurements reached one of these labels the Tracer application was notified of the arrival at this particular landmark by a button press. Based on the trace of button clicks we calculate the average movement speed between two landmarks. To be able to recognize still periods another button was pressed each time the person stopped moving.

The data for the fingerprint database was also collected using the Tracer application. We applied a grid of reference points to the operation area which includes 225 points with a spacing of 1.5 meter (see the blue markers in Figure 3(a)). During fingerprinting, we collected 120 signal strength samples at each reference point, resulting in 27,000 samples in total. For the data collection of the fingerprint database we used the Orinoco network card.

4.2 Feature Analysis

Movement detection using signal strength can be based on several features calculated from a sliding window of signal strength measurements. Previous research has explored the features of Euclidean distance, signal strength variance and rank correlation. The Euclidean distance feature can be calculated in several forms. First, the gap form where the Euclidean distance is calculated between the first and the last signal strength measurement in a sliding window. Second, the average form were the Euclidean distance is calculated between each consecutive signal strength measurement and then average together. Compared to previous work, in this paper we take the assumption that signal strength measurements are only available for one access point to support movement detection with monitor sniffing. This assumption means that the rank-correlation feature can not be used because this feature requires measurements from several access points to rank them in terms of signal strength. The Euclidean distance feature collapses to the absolute difference in signal strength for one access point. The goal of this section is to both analyze how such features behave under movement but also if this behavior can satisfy a number of reliability requirements which are listed in Table 2.

The data collected as described in Section 4.1 allows us to evaluate by means of emulation the mentioned features with respect to these requirements. In the following, we focus on the feature of signal strength variance which is the feature that we later based on our emulation results will choose for our validation of the system. For the analysis Figure 4 to 7 plot the median signal strength variance for each of the 64 different options in our data set within the categories still, slow movement and fast movement. The feature values are calculated using a ten measurements long sliding window and for three access points from the office and for three access points from the hall. These six access points were chosen to be representative out of all measured access points to increase the readability of the graphs. Each single data point in the graph therefore represents the median feature value for a specific access point in one of these 64 options. Within each category the data points are distributed based on when they were collected for readability of the graphs.

A central feature of the ComPoScan system is the use of lightweight monitor sniffing in addition to invasive active scanning. Therefore it is important that the chosen feature will work with signal strength measured using either monitor sniffing or active scanning. In Figure 4 the data points
Table 2: Requirements.
1. Detect using both monitor sniffing and active scanning measurements.
2. Work with different network cards.
3. Work in different physical environments.
4. Work with access points with different physical replacement.
5. Work both at day and at night
6. For people with different mobility patterns.

are marked depending on if they are calculated from monitor sniffing or active scanning measurements. The plotted data confirms that for active scanning measurement the signal strength variance increases with movement as shown by earlier research. However, the plot also shows that a similar increase can be observed for monitor sniffing. Another important consideration for our system is that it should work with different network cards. In Figure 5 the same data is plotted as in the preceding figure but now marked and regrouped with respect to the collecting network card. From the figure it can be noticed that the median signal strength variance change similarly across the categories for all three network cards.

The median signal strength variance not only depend on speed but also the values in the collection area. This also indicates that the signal strength variance not only depend on speed but also the strength of measured access points.

Table 3: Access point statistics showing various signal strength (SS) properties.

<table>
<thead>
<tr>
<th>Access point</th>
<th>Min. SS</th>
<th>Max. SS</th>
<th>Median SS</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-102.0</td>
<td>-55.0</td>
<td>-82.0</td>
<td>6.7%</td>
</tr>
<tr>
<td>2</td>
<td>-97.0</td>
<td>-37.0</td>
<td>-76.0</td>
<td>10.8%</td>
</tr>
<tr>
<td>3</td>
<td>-95.0</td>
<td>-22.0</td>
<td>-56.0</td>
<td>45.2%</td>
</tr>
<tr>
<td>4</td>
<td>-89.0</td>
<td>-26.0</td>
<td>-60.5</td>
<td>13.1%</td>
</tr>
<tr>
<td>5</td>
<td>-100.0</td>
<td>-48.0</td>
<td>-81.5</td>
<td>13.5%</td>
</tr>
<tr>
<td>6</td>
<td>-91.0</td>
<td>-34.0</td>
<td>-62.0</td>
<td>44.9%</td>
</tr>
</tbody>
</table>

The median signal strength variance has also been analyzed with respect to the different mobility scenarios and the time of collection but the graphs for these have been omitted. For mobility scenarios the median signal strength variance showed the same behavior across the categories and had the same spread. In respect to time the same change in signal strength variance could be noticed across categories between day and night data.

The above analysis was based on the median signal strength variance split into three categories. Below we will only consider the two categories of still and moving which are our target categories. Above the median signal strength variance was considered in the following the complete distribution of the signal strength variance is considered for the two categories. Figure 8 plots such distributions for still and moving for the three network cards. This plot shows that the still dis-
4.3 Methods

Several detectors have been applied by earlier work to detect movement using the aforementioned features. Using our data we have evaluated several of these and finally selected a Hidden-Markov Model (HMM) following Krumm et al. [11] as the best option. In addition to the HMM detector we evaluated a simple Naive Bayes detector and two AdaBoost detectors instantiated with respectively ZeroR and Naive Bayes as basic detectors. In the next section, we will provide some emulation results that will support the choice of the HMM. The primary difference between the HMM detector and the three other detectors is that it is able to take previous feature values into account and thereby minimize that the detector is immediately flipping back and forth between detecting movement and detecting still. A drawback with the HMM method is that it is only able to work with one feature type were the others can use several.

The used HMM has two states: still and moving. Probabilities are assigned to each state for staying or transition to the other state. The probability of being in either states are initially set to be equal. Each state also has a distribution associated with it that gives the probability of observing a feature value in this state. In each prediction step a set of consecutive feature values within a sliding window are used by a Viterbi algorithm to calculate the most likely sequence of state changes in the model given a set of feature values. The estimated state is then the ending state of the calculated sequence. The distributions are calculated from a set of training data. In this work the Gaussian kernel method is applied because it creates more generalizable distributions than the histogram method. Based on initial experiments a standard deviation of 0.1 was selected for generating the still distributions and a standard deviation of 1.5 for generating the movement distributions by the kernel method. An important point is that the feature analysis showed that such distributions can be used for different network cards and environments with a different access point availability. This means that the system does not need to be training for each specific deployment which will be further supported by our emulation and validation results.

4.4 Emulation Results

The purpose of this emulation is to identify a good detector for the ComPoScan system and find the parameters for the identified detector that should be used in our validation of the system. An important goal for a detector is that it is good at detecting movement but also that allows us to make different trade-offs to either favor communication or positioning.

Several types of emulations have been run to evaluate the detectors on the collected data. Emulations for Naive Bayes using all features, the HMM with gap Euclidean distance, the HMM with average Euclidean distance and the HMM with signal strength variance have all been run in an extension to the Loceva toolkit [7] implemented by the authors. For the HMM implementation we used the Jahmm library\(^4\). The emulations for AdaBoost using Decision Stumps and AdaBoost using Naive Bayes were run in the Weka toolkit [18]. Initially, emulations were run to find good values for the window size used in the feature calculations and for the history size used by the HMM detectors. The results were that higher values of window size and history size made the detectors better at detecting still but worse at detecting movement. For our data this will improve a detector’s overall accuracy because our data contains more still than moving data. But for the ComPoScan system the overall accuracy is less important than a detector’s ability to detect movement. Therefore we focused

\(^4\)http://www.run.montefiore.ulg.ac.be/~francois/software/jahmm/
on finding a window size and a history size that would make the detectors good at detecting movement without sacrificing to much on the overall accuracy. Another reason for keeping the window size low is that this minimizes the startup time before the system can start making predictions and for the history size it minimizes the computational requirements for the Viterbi algorithm used by HMMs. Based on these criteria we selected a window size of ten and a history size also of ten.

To compare the detectors eightfold cross-validation was applied where data was split into folds depending on the scenario collection round. This makes sure that test and training data have not been collect at the same point in time. For the emulation output movement was chosen as a positive output and still as a negative output. This means that we can count the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). From these counts we can calculate the true positive rate \( tp = \frac{TP}{TP+FN} \) and the false positive rate \( fp = \frac{FP}{FP+TN} \). By varying the parameters of the HMM’s and setting different selection thresholds for Naive Bayes and the two AdaBoost methods a curve of pairs of \( tp \) and \( fp \) can be plotted. Such a graph is known as a Receiver Operating Characteristic (ROC) curve [18]. The curve shows what different trade-offs can be made in terms of \( tp \) and \( fp \). The more a graph for a detector stretches towards the upper left corner the better the detector performs.

The ROC curves for Naive Bayes, the two AdaBoost detectors and HMM with signal strength variance are shown in Figure 10. The curves show that there are only small differences among the different detectors. The AdaBoost detector performs best when the \( tp \) is below 60 while the HMM with signal strength variance performs best above 80. The Naive Bayes detector performs worst regardless of the interval. For ComPoScan we are interested in a detector that is good at detecting movement and therefore we are interested in a detector that can maximize the \( tp \) without increasing the \( fp \) to much. Therefore the HMM with signal strength variance is the best choice. We have also emulated the HMM with the other features and the results are shown focused on the \( tp \) and \( fp \) intervals of interest in Figure 11. Again there are only small differences between the detectors. The HMM with gap Euclidean distance performs best with \( tp \) below 60 but in the interval above 80 which we are interesting in the HMM with signal strength variance performs closely followed by the HMM with average Euclidean distance.

To validate our system we need to fix the parameters for the prototype. For the HMM with signal strength variance the parameters to fix are the two transition probabilities of the HMM. To help with this on Figure 11 two lines are plotted which marks the optimal performance when treating the value of errors in different ratios. The first line treats false negatives and false positives equally and the second threat them in the ratio one false negative to three false positives. On each of the lines we choose one set of parameters as illustrated on Figure 11.

For the parameter set with a probability of 0.011 to change from moving to being still and 0.0011 to change from still to moving we ran further emulations to evaluate how the chosen detector addresses the design requirements listed in Table 2. The emulations were run as cross validation with a different number of folds depending on how many categories they should be split in. Table 4 to Table 7 list the results of these emulations were each entry in the tables are named by the test data. So, the first entry in Table 4 is from an emulation with monitor sniffing data as test data and active scanning data as training data and so on.

The results in Table 4 show that the detector is working equally well using both monitor sniffing and active scanning measurements. Monitor sniffing being a bit better at detecting movement and active scanning best at detecting still. The results in Table 5 highlights that there are some variation across network cards. The TRENDnet network card is best at detecting movement whereas the Centrino card is worst. If we compare these results with the distributions for the different network cards shown in Figure 8 we notice that the TRENDnet distribution for moving is right shifted compared to the other distributions. This means that when testing with TRENDnet on training data from the two other network cards the probability of the detector predicting still is decreased. So, there are some variation across network cards but it is mainly changing the detectors trade-off between predicting still or moving not making the detector unable to detect movement at all.

The results in Table 6 show that in the hall the detector was worse at detecting still compared to the office building. On the other hand in the hall movement was better detected. Comparing the distributions for the two environments in Figure 9 it can be noticed that for the hall the still and moving distributions have a larger overlap than for the office building. This can both be attributed to the absence of walls lowering the variation in signal strength but it can also be attributed to the difference in access points availability. To analyze this claim Table 7 provides results split over the different access points availability.
points. For the weakest and least measured access points one and four we obtain the best still detection rate because of the lower signal strength variance as identified from Figure 7. The results also indicate that overall the access points one to three from the office environment give better still detection results than the access points four to six in the hall environment.

In all the previous results the feature values were calculated from measurements for a single access point. But you can easily extend the calculation of the feature values to use measurements from several access points. With active scanning multiple access points will normally be measured but also sometimes with monitor sniffing measurement will be made to several access points that are on the same channel or on close channels. For signal strength variance we extend the calculation to multiple access points by calculating the average value over the signal strength variance calculated for each single access point. Denoting the number of access points as $k$ we ran simulations with a different size of $k$. The results in Table 8 indicate that both the detection of moving and still improves when increasing $k$. Therefore in our validation we use a detector that at any moment uses the highest $k$ that is possible given the current measurements.

### Table 4: Monitor sniffing versus active scanning.

<table>
<thead>
<tr>
<th>Mode</th>
<th>True positives (%)</th>
<th>False positives (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor sniffing</td>
<td>89.7</td>
<td>21.1</td>
</tr>
<tr>
<td>Active scanning</td>
<td>84.3</td>
<td>25.2</td>
</tr>
</tbody>
</table>

### Table 5: Network cards.

<table>
<thead>
<tr>
<th>Card</th>
<th>True positives (%)</th>
<th>False positives (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Centrino</td>
<td>76.2</td>
<td>12.6</td>
</tr>
<tr>
<td>Orinoco</td>
<td>89.3</td>
<td>35.8</td>
</tr>
<tr>
<td>TRENDnet</td>
<td>93.3</td>
<td>25.4</td>
</tr>
</tbody>
</table>

### Table 6: Physical environments.

<table>
<thead>
<tr>
<th>Environment</th>
<th>True positives (%)</th>
<th>False positives (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office</td>
<td>83.6</td>
<td>20.3</td>
</tr>
<tr>
<td>Hall</td>
<td>89.9</td>
<td>41.7</td>
</tr>
</tbody>
</table>

Using emulation we have also evaluated how ComPoScan impacts position accuracy. The emulation implements ComPoScan’s switching mechanism between monitor sniffing and active scanning. The emulation in each step make a prediction with the chosen detector of being still or moving. The data used in this prediction depend on what was detected in the preceding step. So if moving was predicted in the preceding step the detector uses active scanning data and if still it uses monitor sniffing data. Then in all steps where the state of the switching mechanism is equal to active scanning a position system is allowed to update it’s current position using the active scanning measurements. This emulation is below compared to the results from a positioning system that is allowed to update it’s position in every step. We refer to this system as traditional positioning.

For the positioning system we selected the positioning algorithm that is probably one of the most studied ones: The Gaussian fit probabilistic algorithm proposed in [6]. For the positioning system we applied the fingerprint database as described in Section 4.1.2. To calculate a signal strength distribution for each access point at each reference point we randomly selected twenty measurements out of the 120 previously collected measurements. As already mentioned, only on second floor of the Hopper building a 802.11-based positioning system was deployed. So, all results presented in this section are collected in this part of the building. For the emulation we used the data for the eight scenarios collected in the office building with the Centrino card collecting monitor sniffing measurements and the Orinoco card collecting active scanning measurements.

We compare the emulation results achieved by ComPoScan against the results obtained from traditional positioning. The average positioning error with traditional positioning was 3.81 m and with ComPoScan it was 3.74 m. The results indicate that ComPoScan on average actually improves the position accuracy with 2%. However, for two of the eighth scenarios ComPoScan decreases the accuracy. From the emulation we can therefore conclude that ComPoScan does not have a major impact on the position accuracy and might even improve it in some cases.

### 5. Prototype Implementation

We wanted to see if our ComPoScan system works in real-world deployments in the same way as the emulation results suggest. To be able to deploy the system in many real-world environments, we implemented a prototype that runs on Linux and supports any 802.11 network card that is able to perform monitor sniffs and active scans.

Based on the emulation results we selected only the HMM signal strength variance detector to be implemented. Further, we kept the parameters of this detector easily configurable to make sure that during our validation we can switch between different configurations to trade communication capabilities against positioning accuracy.

The implementation is structured into three different parts that deal with different tasks:

- **Driver**: We wanted the prototype to support at least Atheros-based wireless network cards. The reason for this is that we owned a NETGEAR WG511T network card that contains the Atheros AR5212 chip-set that has not been used for data collection in the emulation. The driver we used for this card is modified in the same way as described in Section 4.1.1.

- **Signal strength measurement system**: Depending on the results calculated by the movement detection system an active scan or a monitor sniff is performed to collect...
signal strength measurements. In case the movement detection system cannot calculate any result (e.g., a lack of sufficient signal strength samples) an active scan is executed. To invoke an active scan or a monitor sniff we used the Loclib library [7]. This library collects signal strength measurements from the kernel driver and makes them available to user-space applications.

- Movement detection system: After a signal strength measurement is collected, the data is stored with other recent measurements in a ten-entries sliding window. For all access points that are available at least in eight of the measurements the signal strength variance is calculated. These values are then forwarded to the HMM to decide whether the mobile device that provided the signal strength measurements is currently moving or not. For the HMM implementation we used the same Jahnmm library. We trained the HMM to detect movement by using the traces we collected in the hall of the Benjamin building.

Furthermore, to be able to evaluate the impact of ComPoScan on communication capabilities and the positioning accuracy of a mobile device we additionally implemented three sub-systems. The so-called network measurement system gauges throughput, delay and packet loss. For this, we utilized a tool called iperf5. Iperf is a client-server application that measures the maximum throughput achievable over a given link. To measure round trip delay and packet loss we implemented a client application that sends out ping requests every 100 ms. The corresponding server application sends back a ping response every time a ping request packet arrives. The identification number contained in each ping request packet is copied into the response. The client is so able to calculate the round trip delay by subtracting the time when a ping request packet carrying a certain identification number was send out from the time when the corresponding ping response packet arrived. In case no response arrives the request or response packet must be lost during transmission.

The second and third sub-systems are required to calculate position estimates based on signal strength measurements. For this, we implemented a signal strength normalization method that makes signal strength data taken from different network cards directly comparable. The method we selected is published by Kjærgaard [9]. It finds a linear function to match signal strength measurements of two different cards by using the least squares analysis. This approach works pretty well and it can easily be applied because only at a few different locations signal strength measurements from the cards that should be normalized are required. However normalization can be completely avoided by using the technique of hyperbolic location fingerprinting proposed by Kjærgaard et al. [10]. On top of this sits the positioning system. As positioning algorithm we selected one of the probably most studied ones: The Gaussian fit probabilistic algorithm proposed in [6].

In Figure 12 the architecture of the ComPoScan prototype implementation is illustrated. The sub-systems belonging to the ComPoScan system are marked by gray boxes. The three additional sub-systems required to evaluate the prototype system are depicted in white boxes. The arrows show how information is distributed between different parts of the overall system.

5 http://dast.nlanr.net/Projects/Iperf/

6. REAL-WORLD VALIDATION

In this section, we present our results obtained from more than a week of prototype deployment. The system is deployed in six buildings of the University of Aarhus, Denmark (see Figure 13). The movement detection accuracy is illustrated from walks through the Hopper, Ada, Benjamin, Stibitz, Shannon and Turing buildings. As the positioning system is only available in the Hopper building the position accuracy measurements are only collected from this place. The network conditions are also investigated showing the benefit of the configurability of the system by using two parameter sets.

Figure 13: This map shows the names of the buildings where the ComPoScan system has been validated in real-world deployments.

For most real-world experiments we configured the HMM detector in such a way that we favored stable network conditions against position accuracy. Or in other words, we wanted to make sure that ComPoScan only performs active scans if it is quite sure that the person in question is moving. If it is not stated otherwise we used the following parameters: the probability to change from moving to being still is 0.0011, the probability to change from being still to moving is 0.011, the window size is 10 and the history size is 10.

6.1 Movement Detection Accuracy

To study the movement detection accuracy of our ComPoScan system we recorded a typical walk a member of the University of Aarhus would go from a office in the Hopper
building to the Cafeteria located in the Benjamin building to pick up a cup of coffee. On the way back the walk contains stops at different locations to chat with colleagues and to pick up mail and printouts. The path leads additionally through the Ada and Turing buildings.

We recorded the walk once in the morning when many people move around in the buildings and once during the night when the building is abandoned. The walk lasts for more than 25 minutes and contains 11 stops. In both walks, the person who walked around is standing still in 79 percent of the time.

... selected positions within the operation area. Based on the least square analysis the parameters are calculated. We also configured ComPoScan to use the second parameter set as defined in Section 4.4. Compared to the previously selected parameter set, this set differently trades movement detection against still period detection. We expect to see the percentage of correctly detected movements to go up and the rate of correctly detected still periods to go down. The comparison of the two parameter sets is based on data we have collected in the Shannon building. This walk contains five stop periods which represent about 67 percent of the more than 13 minutes walk. The results are listed in Table 9.

![Figure 14: Movement accuracy.](image)

The real movement of the two walks is indicated in Figure 14(a). Movement as perceived by ComPoScan is depicted in Figure 14(b). As we see from the two figures, still periods are quite often detected for both walks. The walk performed in the morning shows correctly detected still periods in 87 percent of all real still periods. This is in contrast to 94 percent for the walk during the night. Furthermore, movement is also pretty often detected. However, the correct detection rates are here a bit lower. For the walk at night ComPoScan achieves 54 percent and 65 percent for the walk in the morning. The main reason why ComPoScan detects moving periods better during the day is the following: During the time we performed our walk many people arrived at the office to start working or walked to other offices or meeting rooms to attend meetings. Moving people increase the signal strength variance because each time they walk into the path a radio signal is traveling they attenuate the signal. This means that the signal strength measurements are a bit more broadened. The scattered measurements add to the signal strength variance caused by movement which means movement is easier detectable.

A further analysis of the data shows that there is always a delay between real movement and movement reported by the HMM detector. Three reasons cause this delay: First, if we assume that ComPoScan detected the motionless state correctly then only monitor sniffs are performed. During a monitor sniff only beacons emitted from access points are examined. These results are stored for 600 milliseconds before forwarded to the HMM detector. This procedure is chosen to be similar with an active scan. Even that there might be room for improvements our further analysis shows that this delay accounts only slightly to the overall delay. The second reason is the sliding window we utilize to calculate the signal strength variance. To smoothen variations in the signal strength measurements we applied a window size of ten entries. It takes some measurements to propagate an increased signal strength variance through this window. Third, the history required by the Viterbi algorithm to be able to calculate the most likely sequence of state changes also adds up to the overall delay. For instance, the walk during the night shows an average detection delay of 8.6 seconds with a standard deviation of 13.3 seconds. The minimum delay we observed is only 156 milliseconds and the maximum detection delay is 13.4 seconds. The morning walk shows similar delays.

Additionally, the results presented in Table 9 show also that ComPoScan works in different buildings without any further training. For all our validation walks we only used the data collected in the hall of the Benjamin building to train the HMM detector. To back up our claim that ComPoScan does not require any local training to be functional we went to the Stibitz building for another round of validation. The data reveals that movement is correctly detected in 67.07 percent of all walking periods and still times are correctly detected in 99.23 percent of all cases. This is consistent with the other results we have presented.

In summary, we have shown that ComPoScan is able to work well in different buildings, with different configurations, during different times of a day, and with a new network card.

### 6.2 Positioning Accuracy

As already mentioned, only on the second floor of the Hopper building a 802.11-based positioning system was deployed. So, all results presented in this section are collected in this part of the building.

The positioning system is set up in the same way as described in Section 4.1.3. The parameters for the normalization system are gathered by collecting signal strength samples with both, the Orinoco and NETGEAR network card, at five randomly selected positions within the operation area. Based on the least square analysis the parameters are calculated.
During validation, each time the signal strength measurement system invokes an active scan the measurements are copied and forwarded to the signal strength normalization system. After normalizing the measurements the values are send to the positioning system. The positioning system uses only measurements obtained from one active scan to calculate a position estimate. Before a new position estimate is calculated the positioning system is reset to discard any knowledge learned from a previous measurement. We are aware of tracking technologies might improve the positioning accuracy but we wanted to keep the positioning system as simple as possible to get an clear insight into the impact of ComPoScan on the positioning accuracy.

Signal strength measurements collected from a monitor sniff are omitted because they usually contain only a sub-set of all available access points. While it might be possible to improve the positioning accuracy by using monitor sniffing results we want to investigate how the position accuracy drops if movement is not correctly detected and hence signal strength measurements from active scans are missing.

The Tracer application described in Section 4.1.3 was used to collect real position information of the user during the real-world validation. For this, we sticked labels on the floor to mark prominent places of the path the user was supposed to walk. Based on the Tracer’s trace and known positions of the labels we are able to calculate the real position of the user during the walk. The real position information is than later compared to the position estimates computed by the positioning system.

We selected two 370 seconds walks from our data to investigate the positioning accuracy in detail. During one walk ComPoScan was activated while during the other walk only the positioning system was running. In the latter setup, the positioning system calculates position estimates every 600 milliseconds. We refer to this setup as traditional positioning. Real movement and movement detected by ComPoScan are presented in Figure 15(a) and 15(b), respectively. The real movement for the two walks slightly differ because the person moving around paused a bit more during the ComPoScan walk.

Figure 15(c) depicts the positioning error for traditional positioning as well as the ComPoScan system. On average ComPoScan achieves an error of 4.68 meters whereas traditional positioning is slightly worse by obtaining 4.74 meters. From the graph we see that the curves generated by the two systems look quite similar. During still periods we observe that the positioning error produced by ComPoScan is more stable as what traditional positioning achieves. This is consistent with the average positioning error only calculated for real still periods. ComPoScan achieves 4.06 meters in contrast to 4.16 meters produced by traditional positioning. On the other side, we see from the figure that the ComPoScan position accuracy is worse than what can be achieved by traditional positioning during periods of real movement. On average the positioning error is 8.22 and 8.12 meters, respectively. The reason for this is that ComPoScan calculates only a new position estimate if movement is detected. From the previous section we know that ComPoScan detects movement always bit later than it actually happens. This means if a person already started walking ComPoScan is still perceiving the person to be motionless for a short period of time. Therefore, the increase in the positioning error is caused by this delay.

All our traditional positioning data shows an average positioning error of 4.68 meters. In contrast, the average pos-

Figure 15: Positioning accuracy.
To sum up, we have shown that ComPoScan does not harm the positioning accuracy at all. Even that the positioning error goes up a bit during movement periods it slightly drops during still periods. Further, ComPoScan’s configuration parameters can be used to define it’s sensibility to compute position updates.

6.3 Communication Capabilities

One of the reasons we came up with the idea of ComPoScan is that communication is quite weak in terms of throughput, delay and packet loss if the 802.11 network card is used for positioning at the same time. ComPoScan is build in such a way that it trades communication capabilities against position accuracy. For this it switches to light-weight monitor sniffing when it detects that the person carrying a ComPoScan-enabled mobile device is standing still. During monitor sniffs untroubled communication with the access point the mobile device is associated with is possible. In case ComPoScan detects movement it switches to invasive active scanning to collect enough data to enable the positioning system to calculate an accurate position estimate.

The network measurement system as described in Section 5 is utilized to generate the results presented in this section. We used an extra Apple Airport Extreme access point directly connected to a Fujitsu-Siemens Lifebook T4010 laptop running iperf and the ping server. This setup guaranteed that the only bottleneck is the wireless channel and not the wired network.

We configured iperf to send a UDP stream for 6 minutes with a bandwidth of 17.6 MBit/s. This value was determined by a stepwise increase until no gain could be achieved. The simple UDP transport protocol is selected because the more sophisticated protocols often bring congestion avoidance strategies that may interfere with our measurements. As we are the only participant in this network the UDP measurements are the upper limit of what is achievable. We measured the throughput in the smallest time interval supported by iperf: Every 0.5 seconds. During the time iperf was sending data we measured the round trip delay by using our self-developed application. We configured it to send a ping request every 100 milliseconds.

For the throughput, delay and packet loss measurements we walked around in the Hopper building for 6 minutes. During this walk, we stopped 5 times representing 86 percent of the total time. To be able to compare our ComPoScan system we repeated the walk while performing active scans every 600 milliseconds as a traditional positioning system would request to do.

In Figure 17(a) the real movement for both walks is presented. Figure 17(b) shows how ComPoScan perceives still and movement periods. The correctly detected rate for movement is here 58 percent and 84 percent for being still. The throughput results are depicted in Figure 17(c). ComPoScan is able to transfer 638.3 MBytes during the 6 minutes of the experiment whereas a traditional positioning system reduces the amount of data being transferred to 5.2 MBytes. This corresponds to an improvement of factor 122. On average a throughput of 12.8 MBit/s and 0.1 MBit/s and a standard deviation of 5.9 MBit/s and 0.07 MBit/s is achievable, respectively. During detected still periods ComPoScan is able to transfer 14.9 MBit/s on average and 2.9 MBit/s during detected moving periods. The reason why ComPoScan achieves such a high number during detected movements is the detector update delay. A detector update is available every 600 milliseconds.

6.4 Positioning Capabilities

If we compare the round trip delay, we see a similar picture (see Figure 17(d)). For ComPoScan on average the delay goes down to 46.97 milliseconds during detected still periods and up to 186.36 milliseconds during detected moving periods. This sums up to an total average delay of 53.46 milliseconds. In contrast, traditional positioning achieves only an average round trip delay of 566.58 milliseconds.

Related to the delay are the packet losses. During active scans no data can be transmitted meaning that the data is dropped after the different buffers provided by the network stack of the kernel and the network card driver are filled up. Figure 17(e) shows the packet loss for ComPoScan and traditional positioning during the walks. For ComPoScan the packet loss spikes each time movement is detected (e.g., around 70, 100, and 200 seconds). During these periods ComPoScan drops around 72.79 percent of the packets. Only 3.49 percent of the packets are dropped during detected still periods. On average ComPoScan drops 16.07 percent of all ping request and ping response packets. This is contrast to traditional positioning where 89.78 percent of the packets are dropped. The reason why we see more than 3 percent of packets be dropped by ComPoScan during still periods is how we count packet loss. We consider the state ComPoScan reports during generation of the ping request packet. So, for example, it might happen that a ping request is send out when ComPoScan detects a still period. Directly after the packet left the mobile device ComPoScan recognizes movement and starts an active scan. So, the ping response packet generated by the server application is then dropped because it can not be delivered.

We repeated the validation for the communication capabilities by applying the second parameter set as selected in Section 4.4. This parameter set is supposed to favor position accuracy. The downside of this is that the communication capabilities might be affected. In the following, we investigate how severe the impact is.

Figure 18 shows the throughput for both configurations. The spikes in throughput are not always at the same positions in the graph because the two different configurations influence the movement detection system of ComPoScan differently. Further, the person who validated the system walked with slightly different speeds and stayed a bit longer still at the different places during the two walks. However, as the walks are both equally long in terms of time and contain the same number of still periods and still times, the impact of the two configurations can be compared by looking at the average values over the total walking time.

For the second configuration the average throughput is 10.01 MBit/s with a standard deviation of 10.6 MBit/s. This is on average 2.2 MBit/s less than what we achieved in the previous validation experiment. As shown by the graph, during real still periods ComPoScan using the second configuration switches quite often back and forth. This is the reason why we see such a huge standard deviation. The average delay draws a similar picture: On average the round trip delay is 81.98 milliseconds compared to 53.46 milliseconds drawn from the other configuration. For packet loss, the second parameter set produces a loss rate of 49.62 percent on average. In comparison to the results obtain from the first parameter set this is an increase of 33.55 percent.

To conclude this section, we have shown that ComPoScan generates communication conditions that can be used for
movement

still

0  50  100  150  200  250  300  350

Trad. Pos.  ComPoScan

(a) Real movement

moving

still

0  50  100  150  200  250  300  350

Trad. Pos.  ComPoScan

(b) Detected movement

0  50  100  150  200  250  300  350

Trad. Pos.  ComPoScan

(c) Throughput

0  50  100  150  200  250  300  350

Trad. Pos.  ComPoScan

(d) Round trip delay

0  50  100  150  200  250  300  350

Trad. Pos.  ComPoScan

(e) Packet loss

Figure 17: Communication capabilities.

Figure 18: Throughput for the two different configurations of ComPoScan.

meaningful data transfer. Further, we investigated two configurations to show that ComPoScan can be configured to favor communication capabilities.

7. DISCUSSION

In the literature (e.g., [4], [15]) it is reported that regularly performing active scans consumes more battery power than not scanning at all. For instance, Brunato et al. [4] state that their HP iPAQ H5450 PDA having the 802.11 network card switched off lives for 228 minutes. If the network card is switched on and associated with an access point without sending any data the PDA’s battery is depleted in 140 minutes. In case the PDA continuously scans the lifetime is only 103 minutes. Inspired by these results we investigated the battery lifetime of our IBM R51 laptop while having the network card switched off, scanning actively, and performing monitor sniffing. We followed the recommendations for battery lifetime measurements listed in [3]. Our experiments showed that the 6 cell li-ion battery of our laptop provided energy for 192 minutes if the network card was switched off. In case active scans were performed every 0.6 seconds the laptop lived for 177 minutes. The lifetime is increased to 184 minutes if monitor sniffing is performed. These results show that switching on the network card and selecting different scanning methods matters in terms of battery lifetime. However, the impact is smaller compared to the results shown by Brunato. The reason for this is that the battery drain caused by the different hardware components (e.g., CPU, memory, graphic card) of a PDA is lower compared to laptop. The impact of the network card is relatively smaller for a laptop as for a PDA. However, for resource-constrained devices such as a PDA ComPoScan might also be able to increase their lifetime.

For our movement detection system we heavily rely on monitor sniffing. To be able to perform a monitor sniff the 802.11 network card is configured to work in monitor mode. This mode sets the network card into a listening state that allows to receive frames send from a wireless network the network card is not associate with. Even better, frames from channels close to the channel the network card is using for communication can be retrieved. Nowadays, most available network cards support monitor mode. For instance the Intel Centrino chipsets (e.g., 2100, 2200, 2915, 3945) and nearly all Atheros chip-sets (e.g., AR5002G, AR5004X, AR5005, AR5211, AR5212) as well as older chip-sets such as the Lucent Orinoco chipsets supports monitor mode. Unfortunately, MS Windows does not support monitor mode by the NDIS driver interface which is why many drivers for this operating system do not support monitor mode. On the other hand, Linux and most BSD derivates provide a wide range of drivers with build-in support for monitor mode. If the demand for monitor mode grows we can expect to see more drivers for MS Windows supporting monitor mode as well. Therefore enabling ComPoScan on MS Window is just an implementation issue.

An easy and simple way to mitigate the impact of scanning on communications is to reduce the scan frequency. However, this solution comes with the drawback that the positioning error increases dramatically. For instance, let’s imagine that an active scan is performed only every four seconds. This means that the positioning system is also updated only every four seconds with new signal strength measurements. In four seconds a person can walk up to six meters if we assume a descent walking speed of 1.5 m/s. So, in this scenario on average, three meters have to be added to the positioning error of positioning system. In indoor environments three meters have to be added to the positioning error of the current solution because they distinguish between different rooms. From our point of view, such an approach is not a solution.

Nowadays, modern harddisks as part of laptops often contain accelerometers to protect the drive in case it is accidentally dropped. Many cell-phones also contain such accelerometers to automatically detect if a picture is taken in landscape or portrait orientation. So, the question is can these accelerometers be used to detect movement of a person. Depending on the quality of the accelerometer and how well it
is integrated into a movement detection system the answer is yes (e.g., [12, 19]). Although movement can be detected by this class of sensors high-quality triaxial accelerometers are required and the accelerometers usually integrated into consumer products are only dual-axis ones which do not work as well. However, our system provides the advantage that it works without any additional sensors. This means all the millions of 802.11-enabled mobile devices already deployed all over the world work with our system without any hardware modifications. Furthermore, if accelerometers become omnipresent in mobile devices our system might be extended to make use of them and further improve movement detection.

8. CONCLUSIONS
The primary contribution of this paper is the novel ComPoScan system that can mitigate the effect of scanning on concurrent communications. ComPoScan is based on movement detection to adaptively switch between light-weight monitor sniffing and invasive active scanning. Additionally, we provide an evaluation of the proposed system both by emulation and by validation in a real-world deployment. The emulation showed that our movement detection system works independently of the environment, the network card, the signal strength measurement technology, and number and placement of access points. We also showed that ComPoScan does not harm the positioning accuracy of the positioning system. By validation in a real-world deployment we provided evidence for that the real system works as predicted by the emulation. In addition, we provide results for ComPoScan’s impact on communication where it increased throughput by a factor of 122, decreased the delay by a factor of ten, and decreases the percentage of dropped packets by 75%. Furthermore, as mentioned in the discussion the system is also able to decrease the power consumption.

In our ongoing work, we are trying to address several issues. First, conceive a system that can make further use of monitor sniffing measurements for updating the position estimate without switching into active scanning. Second, evaluate the impact of using accelerometers to implement movement detection. Third, implement our system on a smaller platform (e.g., PDA) that allows us to better evaluate the power savings of our system.

Acknowledgments

The authors acknowledge the financial support granted by the Deutscher Forschungsgemeinschaft (DFG), the European Science Foundation (ESF) and the software part of the ISIS J.Urbinhejberg competence centre.

9. REFERENCES


