Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

*Philippe Bourquin (D-INFK)*

# *Adaptive Sampling for Sensor Networks*

Seminararbeit
Sommersemester 2005

Betreuung: Silvia Santini

Research Group for Distributed Systems

# 1 Introduction

## 1.1 Sensor Networks: Basics

According to the vision of Mark Weiser, that the physical world will be connected with pervasive networks and that everyday devices will be able to sense their relationship to humans and to each other, the ability to sense a broad set of physical phenomena, will become a common aspect of small, embedded devices. Furthermore such devices shall communicate with each other to organize and coordinate their actions. The physical world presents an incredibly rich set of input modalities, including acoustics, image, motion, vibration, heat, light, moisture, pressure, radio, magnetic, etc. Using (wireless) sensor networks it will be possible to observe (i.e. measure) and to sample some of these environmental variables simultaneously at different locations. Basically each sensor node has 3 main tasks:

1. sense
2. compute
3. communicate

Sensor Networks, which are predominantly data-centric rather than address-centric, find application in monitoring environment and activities. Because a sensor network consists of many (hunderds, thousands) nodes, sensor nodes should be as cheap as possible. But using low cost devices results in limited functionality and performance.

A user who wants information from the sensor network can ask queries to the network. These queries are directed to a region containing a cluster of sensors rather than specific sensor addresses. Given the similarity in the data obtained by sensors in a dense cluster, aggregation of the data is performed locally. That is, a summary or analysis of the local data is prepared by an aggregator node within the cluster, thus reducing the communication bandwidth requirements.

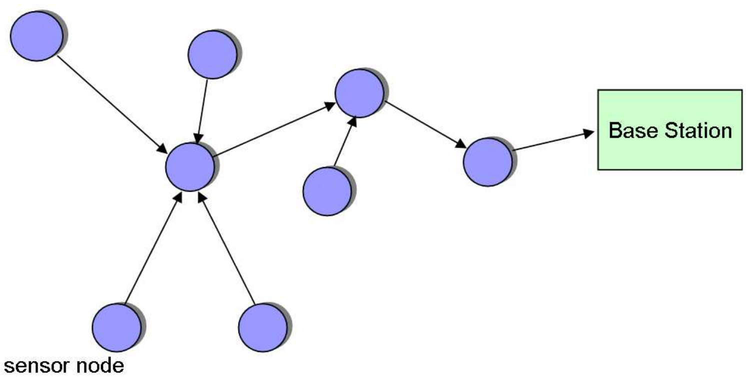

**Figure 1:** Basic Layout of a sensor network. Queries are sent from the base station to the sensor nodes.

Aggregation of data increases the level of accuracy and incorporates data redundancy to compensate node failures. A network hierarchy and clustering of sensor nodes allows for network scalability, robustness, efficient resource utilization and lower power consumption.

## 1.2 The Environment

Natural environments are often extremely dynamic and therefore sensors will need to continously adjust to dynamic systems. The challenge is to represent an accurate picture of the changes in the environmental variables. This can only be achieved if the physical phenomenon is sensed or sampled from the environment at an accurate rate. The physical phenomena measured ultimately dictates spatial and temporal sampling scale. High-frequency waves typically require higher temporal and spatial sampling than phenomena such as temperature or baromentric pressure, where spatial and temporal fluctuations are coarser-grained. The sampling rate is a function of both the phenomena and the application.

## 1.3 Sensor network properties

Since sensor networks differ from traditional distributed systems, hardware (sensor nodes) and software (algorithms) must be adapted and take some special properties of sensor networks into account. Focusing adaptive sampling algorithms, some of the most important properties are listed below:

### 1.3.1 Density

In future applications sensor networks are envisioned to consist of hundreds or even thousands of single nodes, which all communicate with each other through an ad-hoc wireless network. At certain places nodes can easily reach a high density. System density is a measure of sensor nodes per footprint of input stimuli. Higher-density systems provide greater opportunities for exploiting redundancy to eliminate noise, improve accuracy and extend system lifetime.

### 1.3.2 Self-organization

Sensor nodes organize themselves in an ad hoc fashion after the deployment. Since physical access to the nodes will normally be extremely limited, sensor nodes should work lifelong without human intervention. The sensor networks automatically detects when a node crashes or runs out of batteries and re-configures itself.

### 1.3.3 Ressource constraints

Generally, we assume that each node in a wireless sensor network has certain constraints with respect to its energy source, power, memory, storage, and computational capabilities. Not only the ressources of the single sensor nodes are limited, but also those of the network as a whole. Especially in wireless sensor networks, which have one shared medium and therefore have to deal with packet collisions, the network capacity is strongly limited.

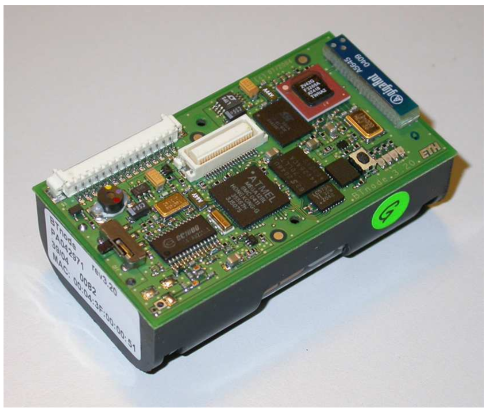

**Figure 2:** Example of a sensor node: BTNode.

| | |
|---|---|
| Microcontroller: | Atmel ATmega 128L (8 MHz @ 8 MIPS) |
| Memories: | 224 Kbyte RAM, 128 Kbyte FLASH ROM, 4 Kbyte EEPROM |
| Communication: | Bluetooth, Zeevo ZV4002 or low power radio (868MHz) |
| Programming: | Standard C, TinyOS compatible |

**Table 1:** Specification of BTNode Rev. 3

## 1.4 Motivation for Adaptive Sampling

Given a set of network and environment characteristics and definitions, ressource consumption (energy and network bandwidth) should be minimized while maximizing the measurement accuracy. The aim is to produce an accurate spatial picture of a certain physical process, while making an efficient use of resources As events are not uniformly distributed in the environment, not all sensor nodes sould collect data samples at a common, fixed rate. Let us look at an example [6] to illustrate this issue:

Habitat monitoring is a very well-known scenario for sensor network applications, and a first concrete experiment in this field was carried out on the Great Duck Island. Sensors were deployed in burrows of Storm Petrels (a seabird) for monitoring purposes. During the day time, the burrows were expected to be empty, an thus a low sampling rate should be sufficient (avoid idle listening). However, if some unusual measurements are recorded at some burrows, it would be desirable to collect samples from them more frequently than from other burrows. This can be achieved by adaptive sampling.

# 2 Adaptive Sampling

## 2.1 Sampling: Basics

Most physical phenomenas are widespread in time and space. Thus, they can be analyzed in one or both these domains. For example a certain point in space can be considered and the time dependency of the signal at this point is analyzed. However, there also exists a space domain, which can only be examined with multiple sensors at different positions in space. A lot of work in adaptive sampling has be done in both the time- and space domain. In this study we focus on the time domain.

### Time domain

In order to properly sample an analog signal the Nyquist-Shannon sampling theorem must be satisfied. In short, the sampling frequency must be greater than twice the bandwidth of the signal (provided it is filtered appropriately):

$$f_s \leq 2B_n \tag{1}$$

$f_s = sampling\ frequency$
$B_n = network\ bandwidth$

Figure 3 illustrates how the sampling interval is defined. With adaptive sampling the sampling interval may vary over time and adapts to the signal complexity. But how can the sensor nodes determine the correct sampling rate? Should a central server take part in this decision process? Following, 3 different possibilities for a adaptive sampling approach in sensor networks are introduced:

- **"God view":** A server knows everything (signal complexities at all sensor positions) and communicates the sensor nodes the appropriate sampling rate.
- **Completely autonomous nodes:** Each sensor node determines on its own the correct sampling rate.
- **Partially autonomous nodes:** Sensor nodes communicate with server only from time to time and adapt sampling rate most of the time autonomously.

## 2.2 Oversampling

When measuring a signal, a minimal accuracy should always be guaranteed. This can easily be achieved by using the naive solution of oversampling. An oversampled signal can be defined as a signal, which is sampled at a higher rate than necessary for the representation of the signal bandwidth. [wikipedia]

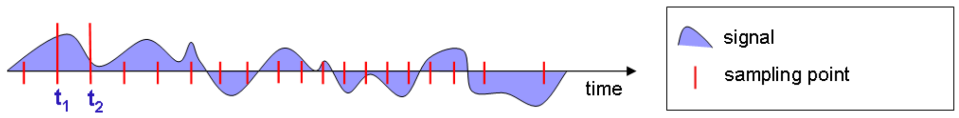

**Figure 3:** Example of a 1-dimensional signal and possibility how to sample it. The time between two neighboured sampling points (e.g. $t_1$ and $t_2$) is called sampling interval.

**Problems with oversampling**

Sensor nodes using oversampling will run into several problems:

- Most of the sampled data must be transmitted to a central server, the base station (see 1). The CPU at the central server might have to process unneccessary data from numerous sources, but this would not affect the result significantly.
- Uneccessary data is transmitted through the communication channel. Moreover, in cases of low bandwidth networks the option of oversampling might not be available at all.
- Power conservation is critical for sensor nodes. Oversampling leads to increased power consumption because of increased data transmission.

Thus, oversampling is not a suitable solution for sensor networks. The goal of an adaptive sampling approach is to make the rate of sensing dynamic and adaptable: As signal complexity grows, the sampling interval is scaled down and vice-versa. Two different approaches for realizing adaptive sampling for sensor networks are introduced in the following part: The Feedback Control Mechansim and the Kalman Filter.

## 2.3 "God view"

The "God view" approach mentioned can only be realized if a central server has the complete information about the network and the environment. In other words, the server must know the signal complexity at each sensor node. This is impracticable without prior communication with the nodes. Furthermore the network bandwidth would be heavily reduced by many "sampling instruction packets" sent out by the server. Therefore, such an approach can not be taken into account.

## 2.4 Completely autonomous nodes: Feedback Control Mechanism

Each individual node adopts a feedback control mechanism in order to make the rate of sensing dynamic and adaptable. Figure 4 presents the feedback control mechanism proposed in [7]. With an internal model representing the environment, each node can make predictions of future measurements. The real sampled data will get compared to these model predictions and an error value will be calculated on the basis of the comparison. If the error value is more than the predefined margin of error, then the node will collect the data at higher sampling rate, and if it is lower, the sampling rate will be decreased. Each node decides on its own how to adapt the sampling rate. The adaptation works completely autonomous.

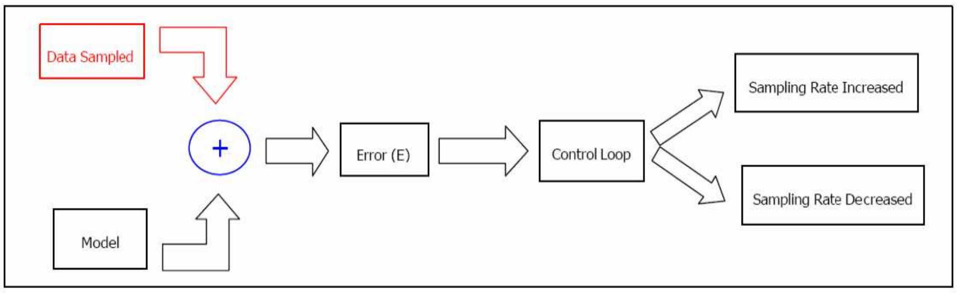

**Figure 4:** Control mechanism used internally within the nodes to control the sampling rate.

### 2.4.1 Experiments and preliminary results

An agent-based simulator was used to test the proposals and the following experiment was set up. An individual sensor was placed in an environment where the temperature is verying by a constant sinusoidal. So in this case the sensor is interested in tracking the temperature. The internal model of the node is a straight line, given by the equation $y_p = ax + b$, where $y_p$ is the predicted tempereature and $x$ is the time at which the temperature occurs. The nodes measure the temperature of the environment every so many units of time (epochs), and as soon two measured values are available to the sensor node, it uses the model equation to calculate the temperature of the environment until the next time the sensor makes a reading from the environment. Then the actual temperature will be compared with the prediction in order to calculate an error value. The experiment was repeated several times with different sinusiodial frequencies and sampling rates. Figure 5 presents the results of these experiments. As it was predicted the error was behaving in a sinusoidal manner. It is apparant that for the same error to be achieved a higher sampling rate is requiered the higher the frequency of the sinusoid.

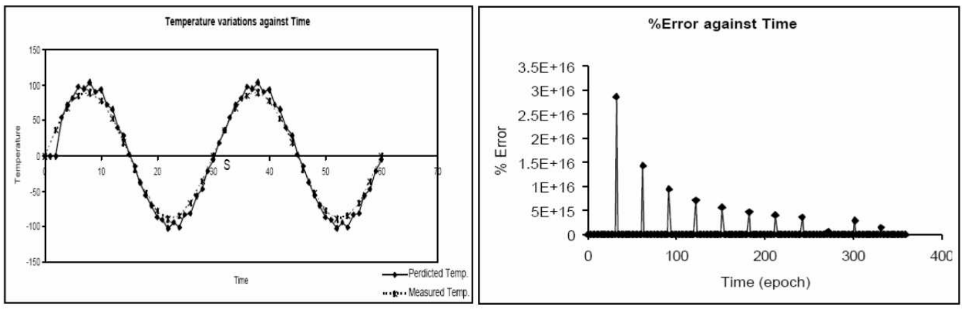

**Figure 5:** Left: Temperature variations against time. Right: Error calculated between predicted and measured temperature.

## 2.5 Partially autonomous nodes: Nodes with Kalman Filter

The approach proposed in [6] employs a Kalman-Filter-based estimation technique wherein the sensor nodes can use the Kalman Filter estimation error to adaptively adjust its sampling rate within a given range, autonomously. When the desired sampling rate violates the range, a new sampling rate is requested from the server.

### 2.5.1 The Kalman Filter

The Kalman Filter is a linear algorithm taht estimates the internal state of a system based on a prediction/correction paradigm. The system model is represented in the form of the following equations:

$$x_{k+1} = \phi_k x_k + \omega_k \tag{2}$$

$$z_k = H_k x_k + \nu_k \tag{3}$$

$x_k = state\ vector\ of\ the\ process$
$\phi_k = state\ transition\ matrix\ relating\ x_k\ to\ x_{k+1}$
$\omega_k = process\ model\ noise$
$z_k = measurement\ model\ noise$
$H_k = matrix\ relating\ system\ state\ and\ measurement\ vector$
$\nu_k = measurement\ noise$

$k = discrete\ time\ index$

The advantage of using the Kalman Filter is that it gives satisfactory results even when we cannot model the process accurately. Error estimates can be further improved using more sophisticated solutions like Praticle Filter or condensation as they work on non-Gaussian noise processes and multi-modal state propagation. Such algorithms are likely to provide better results, but this performance upgrade comes at increased cost of computational resources.

### 2.5.2 Sensor side module

Let $SI_i$ denote the current sampling interval at sensor node $i$. The sampling interval is the inverse to the sampling rate. Let $SIR_i$ denote the range within which the sampling interval can be adjusted by the source without any server mediation and $SI_i^{last}$ denote the last sampling interval received form the server and $SI_i^{desired}$ is desired sampling interval based on the Kalman Filter prediction error. Sensor node $i$ need not to contact the server provided that

$$(SI_i^{last} - SIR_i/2) \leq SI_i^{desired} \leq (SI_i^{last} + SIR_i/2).$$

In this case, $SI_i$ directly takes the value of $SI_i^{desired}$. Otherwise, if the equation above is not satisfied, a new sampling interval is requested from the server. In addition for each request of decrease in the sampling interval, the sensor node sends the fractional error $f_i$ to the server.

### 2.5.3 Server side module

The server manages bandwidth allocation. The allocation algorithm is executed each time a request for a change in sampling interval is received from a sensor node. The server maintains a variable $R_{avail}$ that holds the amout of communication resource available at any time. When a sensor node reports about an increase in its sampling interval (decrease in the sampling rate), the server adds the proportional amount of resource units to $R_{avail}$ and sends an acknowledgment to the sensor node. When a request for a decrease in a sampling interval is received, the request is added to a job-queue that is processed continuously by a separate thread. Each job $J_p$ in the job queue has 5 attributes:

1. *Fractional error* $f_p$ is received from the sensor node when it sends a request.
2. *Request* $Req_p$ is the units of resources requested.
3. *History* $h_p$ is the age of the request in the job-queue. Its value is incremented each time the job queue is processed.
4. *Grant* $g_i$ is the fraction by which, the $Req_p$ has been satisfied so far.
5. *Query weight* $w_p$ the weight of the streaming source from the query evaluator.

We can formulate a linear optimization problem, minimizing the total error over all jobs. If the server allocates $A_p$ units of resources, then the residual error after satisfying the job is proportional to $(1 - A_p/Req_p)$, and the objective function can be formulated as:

$$\min_{A_p} \left( \frac{f_p}{\sum f_p} * \frac{h_p}{\sum h_p} * \frac{w_p}{\sum w_p} * \frac{g_p}{\sum g_p} * (1 - \frac{Ap_p}{Req_p}) \right) \qquad (4)$$

with the 2 constraints

$$\sum A_p \leq R_{avail} \tag{5}$$

$$0 < A_p \leq Req_p \tag{6}$$

Once the optimization problem is solved, the resource units are distributed to the requesting sensor nodes and the job-queue attributes are updated accordingly.

### 2.5.4 Results

Like the Feedback Control Mechanism, the Kalman Filter approach was tested by computer simulation. The performance of the system was evaluated based on an *effective resource utilization* (ERU) metric $\xi$, which is calculated as $\xi = \eta * m$ where $m$ is the fraction of messages exchanged between sensor node and server, to the total number of tuples read by the sensor node, where $\eta$ is the mean fractional error between the actual trajectory and that generated by interpolation. Results are presented in Figure 6.

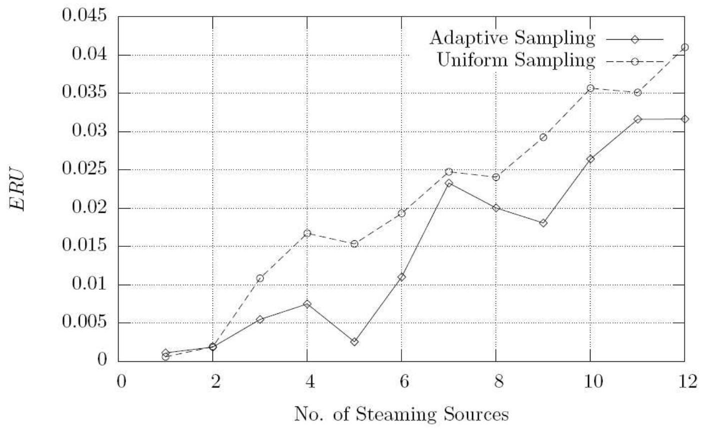

**Figure 6:** ERU on varying number of sensor nodes.

## 3 Conclusions and Future Work

In this work it was showed how the properties of sensor networks and those of the environment (e.g. irregular distribution of events in time and space) can juggled. Resources in sensor networks are strongly limited and thus resource consumption (energy, network bandwidth) must be minimized. Adaptive sampling handles this issue by making the rate of sensing dynamic and adaptable to the signal complexity of the environment. While trying to maximize accuracy, resource consumption is minimized. Two possible approaches for implementing adaptive sampling were introduced: The Feedback Control Mechanism and the Kalman Filter.

| **Feedback Control Mechanism** | **Kalman Filter** |
|---|---|
| - no communication overhead | - most of the time no communication (hopefully) |
| - fast adaptation possible (no acknowledgment to wait for) | - network performance optimized |
| - no additional requests to server necessary | - server side module necessary |

### 3.1 Future Work

Both presented approaches for adaptive sampling were only tested by computer simulations. In the real world the results could be affected by other aspects, such as collision behavior or physical obstacles between sensor nodes. Further research could be done in following directions:

- Extending current architecture to multi-hop sensor networks.
- Developping algorithms to incorporate adaptive $SIR$s in the Kalman Filter approach.
- Testing the system performance on more real life data sets.
- Developping algorithms that can do adaptive sampling in the space domain.
- Testing behavior on node failures and on adding new nodes to the existing network.

## References

[1] P. Bonnet, J. E. Gehrke, and P. Seshadri. Querying the physical world. *IEEE Journal of Selected Areas in Communications*, 7(5):10–15, October 2000.

[2] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. *IEEE Pervasive Computing*, 1(1):59–69, 2002.

[3] ETH Zürich (D-INFK, D-ITET). Btnodes - a distributed environment for prototyping ad hoc networks. Available at http://www.btnode.ethz.ch.

[4] D. Ganesan, S. Ratnasamy, H. Wang, and D. Estrin. Coping with irregular spatio-temporal sampling in sensor networks. In *2nd Workshop on Hot Topics in Networks (HotNets-II) 2003*, Cambridge (MA) USA, November 2003.

[5] Intel Research Laboratory at Berkeley, University of California at Berkeley, College of the Atlantic. Habitat monitoring on great duck island. Available at http://www.greatduckisland.net.

[6] A. Jain and E. Y. Chang. Adaptive sampling for sensor networks. In *Proceeedings of the 1st international workshop on Data management for sensor networks (DMSN '04)*, 2004.

[7] A. D. Marbini and L. E. Sacks. Adaptive sampling mechanisms in sensor networks. 2003.