Sensor Network Data Fault Types

KEVIN NI, NITHYA RAMANATHAN, MOHAMED NABIL HAJJ CHEHADE, LAURA BALZANO, SHEELA NAIR, SADAF ZAHEDI, GREG POTTIE, MARK HANSEN, and MANI SRIVASTAVA
University of California, Los Angeles

This tutorial presents a detailed study of sensor faults that occur in deployed sensor networks and a systematic approach to model these faults. We begin by reviewing the fault detection literature for sensor networks. We draw from current literature, our own experience, and data collected from scientific deployments to develop a set of commonly used features useful in detecting and diagnosing sensor faults. We use this feature set to systematically define commonly observed faults, and provide examples of each of these faults from sensor data collected at recent deployments.

Categories and Subject Descriptors: B.8.1 [Reliability, Testing, and Fault-Tolerance]: Fault tolerance and diagnostics, Learning of models from data; C.4 [Performance of Systems]: Fault tolerance, Reliability, availability, and serviceability

General Terms: Reliability, Design

Additional Key Words and Phrases: Data Integrity, Fault, Sensor Network

1. INTRODUCTION

Sensor networks provide us with information about phenomena or events at a much higher level of detail than previously available. In order to make meaningful conclusions with sensor data, the quality of the data received must be ensured. While the use of sensor networks in embedded sensing applications has been accelerating, data integrity tools have not kept pace with this growth. One root cause of this is a lack of in-depth understanding of the types of faults and features associated with faults that can occur in sensor networks. Without a good model of faults in a sensor network, one cannot design an effective fault detection process.

The purpose of this tutorial is to provide a systematically characterized taxonomy of common sensor data faults. We define a data fault to be data reported by a sensor that is inconsistent with the phenomenon of interest’s true behavior. As discussed in section 2, faults are very common in sensor network deployments, and this impacts the ability of scientists to make meaningful conclusions. Examining the large amounts of data from sensor network deployments available at the Center for Embedded Networked Sensing (CENS) as well as other institutions, we have selected datasets that represent the most common faults observed in a deployment. We use these datasets as examples to support the selection and characterization of the faults.

By providing a list of the most commonly seen faults, the material presented here can be utilized in several ways to make a sensor network more robust to faults. In the initial design stage of a sensor network system, the designer could account for and anticipate such faults so that negative impact of a fault can be reduced. When testing a fault detection system, because the faults presented here are the most common, they should be the first to be used in testing by injecting them into...
either simulated or real datasets. For a system that has been deployed, this list can be used as a first step to screen data for common faults and possibly fix sensors. Finally this list can be used to establish a standardized method of evaluating fault detection and diagnosis algorithms for embedded networked sensing systems.

In order to systematically define faults, we also present a list of the most commonly used features in practice to model both data and faults. We use the term features to generally describe characteristics of the data, system, or environment that can cause faults or be used for detection and be modeled. The models based upon these features describe either the expected behavior of the data or the typical behavior of a fault along a set of feature axes. We do not provide a full algorithm for use in detecting any particular fault. However, to show the utility of certain features, we will provide simple examples where they have proved to be useful in practice.

2. PRIOR AND RELATED WORK

Sensor faults have been studied extensively in process control [Isermann 2005]. Tolerating and modeling sensor failures was studied in Marzullo [1990]. However, studying faults in wireless sensing systems differs from faults in process control in a few ways that make the problem more difficult. The first issue is that sensor networks may involve many more sensors over larger areas. Also, for a sensor network the phenomenon being observed is often not well defined and modeled resulting in higher uncertainty when modeling sensor behavior and sensor faults. Finally, in process control, the inputs to the system are controlled or measured, whereas in sensing natural phenomena this is not the case.

As sensor networks mature, the focus on data quality has also increased. The many deployment experiences shows that this is a major issue that needs to be addressed. With the goal of creating a simple to use sensor network application, Buonadonna et al. [2005] observe the difficulty of obtaining accurate sensor data. Following a test deployment, they note that failures can occur in unexpected ways and that calibration is a difficult task. Using this system, Tolle et al. [2005] deployed a sensor network with the goal of examining the microclimate over the volume of a redwood tree. The authors discovered that there were many data anomalies that needed to be discarded post deployment. Only 49% of the collected data could be used for meaningful interpretation.

In a deployment at Great Duck Island, Szewczyk et al. [2004b] classified 3% to 60% of data from each sensor as faulty. Also, Werner-Allen et al. [2006] take a “science-centric” view and attempt to evaluate effectiveness of a sensor network being used as a scientific instrument with high data quality requirements. They evaluate a sensor network based upon two criteria, yield and data fidelity, and determine that sensor networks must still improve.

Now we examine several existing fault detection methods; we discuss the major assumptions and the fault models upon which the detection methods are focused. We also discuss some areas which may benefit from having a systematic fault definition. We see several features for specific faults that are defined that we will incorporate when defining our fault taxonomy.

Elmahrawy and Nath [2003] identify two main sources of errors, systematic errors
creating a bias, and random errors from noise, but focus on the latter. Identifying several sources of noise, they attempt to reduce the uncertainty associated with noisy data using a Bayesian approach to clean the data. The sensor noise model assumed is a zero mean normal distribution, and prior knowledge comes in the form of a noise model on the true data. With a more accurate real world sensor model, the sensor noise model and the prior noise model may be improved.

Deshpande et al. [2004] use models of real-world processes based on sensor readings to answer queries to a sensor network for data. Using time-varying multivariate Gaussians to model data, the authors respond to a predetermined set of query types, treating the sensor network like a database. To some extent this shields the user from faulty sensors. However, the authors point out that more complex models should be used to detect faulty sensors and give reliable data in the presence of faults.

Many of the recent fault detection algorithms have either vaguely defined fault models or an overly general fault definition. Koushanfar et al. [2003] briefly list selected faults, and develop a cross validation method for online fault detection based on very broad fault definitions. Briefly describing certain faults, the authors target transient, “soft” failures, [Mukhopadhyay et al. 2004] using linear auto-regressive models to characterize data for error correction. The errors are modeled only as inversion of random bits which become the focus of local error correction. In Jeffery et al. [2006], the authors attempt to take advantage of both spatial and temporal relations in order to correct faulty or missing data. By defining temporal and spatial “granules,” the authors require the assumption that all data within each granule are homogeneous. Readings not attributable to noise are considered faults.

Additionally, Elahrawy and Nath [2004], Ni and Pottie [2007], and Krishnamachari and Iyengar [2004] exploit spatial and temporal relations in order to detect faults using Bayesian methods. Elahrawy and Nath [2004] introduce a method of learning spatio-temporal correlations to learn contextual information statistically. They use Markov models and assume only short range dependencies in time and space, i.e. the distribution of sensor readings is specified jointly with the readings of immediate neighbors and its own previous reading. The Bayesian approach is also evident in Krishnamachari and Iyengar [2004]. However, their sensor network model assumption of having massively over-deployed sensor networks is not applicable in the type of sensing applications we target. Also their fault recognition assumes any value exceeding a high value threshold is a fault, which may not always be the case.

In Ni and Pottie [2007], it is assumed that sensors only need to be correlated and have similar trends, and a detection system based upon this assumption is developed. The authors use regression models to develop the expected behavior combined with Bayesian updates to select a subset of trusted sensors to which other sensors are compared. There is limited success in modeling mainly due to the lack of a good fault model and a good way of modeling sensor data.

An experiment involving sensors deployed in Bangladesh to detect the presence of arsenic in groundwater cites the importance of detecting and addressing faults immediately [Ramanathan et al. 2006]. The authors develop a fault remediation
system for determining faults and suggesting solutions using rule-based methods and static thresholds.

A key source of error in sensor networks is calibration error. Sensors throughout their deployed lifetimes may drift, and it is important to correct for this in some manner. In Buonadonna et al. [2005], calibration is performed offline before and after a sensor network deployment. The authors determine that calibration is a difficult challenge for future development. Both Bychkovskiy et al. [2003] and Balzano and Nowak [2007] suggest methods to perform calibration online while the sensor network is deployed without the benefit of any ground truth readings.

The initial work of Bychkovskiy et al. [2003] uses a dense sensor deployment with the assumption that all neighboring sensors should have similar readings. However, sensor networks in use do not have the type of dense deployment assumed in the paper. Balzano and Nowak [2007] remove this assumption and use the correlation between sensors to determine the calibration parameters of an assumed linear model. While both of these works have moderate success in applying their algorithms to actual data, they recognize that the lack of good knowledge of true sensor values is a major handicap.

Sheng et al. [2007], focusing on a single fault type, seeks to detect global outliers over data collected by all sensors. They estimate a data distribution from a histogram to judge distance based outliers. The authors have a well defined fault model based on distance between points.

Looking beyond fault detection and correction techniques, there has been relevant work that frames our thrust to provide a fault taxonomy.

Following sensor network deployments, both Szewczyk et al. [2004] and Ramanathan et al. [2006b] explore likely causes for errors in data and node failures for their specific deployment context. While Szewczyk et al. [2004] focus mainly on communication losses, the authors also cite causes for abnormal behavior by certain types of sensors. Ramanathan et al. [2006b] focus on the specific case of a soil deployment where sensors are embedded at various depths in the soil monitoring chemical concentrations. The authors determine the specific hardware issues that caused the faulty data. Both of these works focus on the causes of abnormal data patterns in their respective applications but do not systematically characterize the resultant fault behavior.

Features for use in assessing data quality are explained in Mourad and Bertrand-Krajewski [2002] and exploited in an urban drainage application in Bertrand-Krajewski et al. [2003]. The focus of these works is data validation using their defined features. We will expand on these ideas and move beyond their specific application in order to model all types of faults.

Sharma et al. [2007] focus on a small set of possible sensor faults observed in real deployments. Three types of faults are briefly defined, and different methods of detecting faults are examined. Then, three collected data sets from sensor deployments are analyzed to determine the efficacy of these fault detection methods. We will more clearly define the faults presented and will generalize their definitions to more application contexts.
3. DATA MODELING AND FAULT DETECTION

We first discuss some fundamentals of sensor network design and data modeling to put into context where one can utilize this fault taxonomy. There are two basic applications of sensor networks, environmental monitoring and event detection. In environmental monitoring, which is our primary focus, data is constantly collected and utilized in scientific or other applications. However, in event detection, one is only interested in detecting the occurrence a specific or an “interesting” event [Gupchup et al. 2007].

While our primary focus is on environmental monitoring and the data collection involved, these faults and our framework are still applicable to event detection sensor networks. In the environmental monitoring application, a fault is anomalous data that exceeds normal expected behavior. In event detection, both events and faults present themselves as anomalous data that exceeds normal expected behavior. Thus, what is defined as a fault on our list, may characterize an event. To differentiate between a fault and an event, a training phase may be utilized to determine a model of a specific interesting event. Without such a model for events it is impossible to determine whether anomalous behavior is a fault or an event without human intervention.

In either application, better hardware and software can potentially reduce faults, but at greater financial or computational costs. Even when better hardware is available, this hardware will still likely produce faulty data if only because it can now be used in more challenging environments to answer more difficult questions. For example, section 5.2.4 presents an example of a costly ISUS nitrate sensor in a controlled calibration environment that will produce unreliable data at higher nitrate concentrations.

3.1 System Assumptions

There are a wide variety of assumptions made on both the sensor network and the data for the fault detection algorithms in the presented literature. However, there are a few common assumptions to most of the systems that we will also make. The first assumption is that all sensor data is forwarded to a central location where the data processing occurs. This is conceptually simpler and more convenient, as we do not require any type of distributed computing algorithm for statistical computations.

We recognize that local processing may occur to reduce overall communication costs. However, by the data processing inequality [Cover and Thomas 1991], with more local processing it is likely that less information is available at the fusion center, which may likely result in lowered confidence in fault decisions. Therefore, our discussion represents a best-case scenario, and we will not address the trade-off between decentralization and data quality loss.

The next assumption we make is that all data received by the fusion center is not corrupted by any communication fault. In order to keep things simple, missing data, which may be due to a communication error or not, is simply treated as data not collected and not as a sign of fault. The alternate view of missing data as a sign of a sensor fault has merit in certain cases where data is expected at regular intervals such as the heartbeat messages in Werner-Allen et al. [2006]. This is not
the focus of this paper, as we are only concerned with data faults.

Finally, we also assume that we do not have malicious attacks on the sensor network system. While there has been much work in the security in sensor networks [Shi and Perrig 2004], this is beyond the scope of this work.

3.2 Sensor Network Modeling

Modeling data is the basis for all fault detection methods, and we emphasize its role here. All the work presented here on fault detection techniques employs models, and this is either explicitly stated or generally assumed. We define a model to be a concise mathematical representation of expected behavior for both faulty and non-faulty sensor data. A model may define a range within which data is expected to be, or it may be a well-defined formulaic model. A formulaic model should be able to generate simulated data and faults that behave similarly to the expected true phenomenon.

Data modeling is vital because, in the likely absence of ground truth, faults can only be defined relative to the expected model. By developing a set of models with which data is to be compared, data can be classified as either good data or as belonging to a particular type of fault. As we will see in section 3.3 and noted in El-Nahrawy and Nath [2003], the models developed are heavily dependent on the sensor network deployment context and phenomenon of interest as they can alter the interpretation and importance of certain faults.

Human input is a necessary component in modeling and system design, providing vital contextual knowledge for modeling expected behavior and faults. By selecting the features of importance to the application, humans are better able to incorporate contextual information into models than any automated algorithm.

If models do not fit the data within a given confidence level, human input can be used to create new fault models, validate unusual measurements, and/or update the accuracy of the models. The initial set of models may be incomplete; models may not be complex enough to capture features that humans did not notice before. However, as we learn more about the phenomena at the scales at which we are measuring, our models will be updated and improved; as such, the need for human involvement should decrease, but never disappear, as the system develops.

3.3 Fault Detection System Design

This list of faults can be utilized in several ways when designing a fault detection system, as they serve as a basis of what to expect in a real deployment. These faults and features can be incorporated a fault detection system to more accurately identify faulty data. To test the efficacy of a detection system, faults may be injected into a test data set. As our list provides for the most common faults in a sensor network, the listed faults can be the first ones to be tested.

When analyzing data from a deployed sensor network, anomalous data can be first checked against this list of faults in an automated manner to eliminate the simple cases and simple causes. This reduces the workload for data users by leaving unclassified anomalous data for analysis, e.g. in updating the fault models.

The application for which the sensor network is being used and types of sensors used play an important role in the design of a fault detection algorithm. Assumptions for one application or sensor may not hold true in another, e.g. the day to
Day variations for soil CO₂ concentrations of figure 7 are not expected to be the same as the light intensity of figure 12. Because of this, there is generally no single module that can detect a particular fault regardless of sensor type.

The most common major classes of sensors have been used extensively in environmental monitoring deployments are temperature, humidity, light (including photosynthetically active solar radiation sensors), and chemical. There are other sensor types that are used more for event detection that are not covered here, e.g., seismic and acoustic sensors. The faults listed are very generalizable to different classes of sensors because each fault can potentially occur on any sensor type. Some sensors that will be more likely to exhibit certain faults than others. While sensor class plays a role in the frequency of faults, sensor specifications are a major influence on the frequency of faults.

4. SENSOR NETWORK FEATURES

*Features* is a general term used to describe characteristics of the data, system, or environment that can cause faults or be used for detection of faults. To systematically define and model faults, we detail a list of features that have been commonly used and presented in the literature. From this list, we select features that are most relevant to each particular fault. These features will also be used to better understand the underlying causes for faulty behavior. While not an exhaustive list of all possible ways to describe data, it is sufficient for sensor network data in particular.

To systematize our taxonomy, we categorize features into three classes, also referred to in Mourad and Bertrand-Krajewski [2002]: *environment features* derived from known physical constants and expected behavior of the phenomenon, *system features* derived from known component behavior and expected system behavior, and *data features*, usually statistical, calculated from incoming data. All three of these feature types are interdependent and influence each other. For example, Szweczyk et al. [2004] discuss how the environmental effect of rain may cause a short circuit on the sensor board that manifests itself in the data with abnormal readings.

Dependent upon the context of a feature description, *features* can be the cause of the fault, can be used to describe or identify a fault, give the context of a fault, or define the location of a fault. We will clarify how the term *feature* applies in particular contexts.

One feature that is not listed in the categories below is time scale. Modeling the expected behavior over only recent data samples, i.e. windowing, is done frequently in the literature for online detection systems. Because the duration of the fault has bearing on its detection and diagnosis, the window size for time-dependent features such as the moving average should be selected according to the sensing application. The window size may be selected from human expertise or by optimizing a specific model quality metric such as mean square error as in Ni and Pottie [2007].

For each feature class, we provide a table defining and summarizing each feature within that class. We also provide additional details of each feature with some examples and how some can have an effect on faults or fault detection in the accompanying text.
Table I. Sensor Network Environment Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Definition</th>
<th>Examples of Features or Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Location</td>
<td>GPS location, ((x,y,z)) coordinates, or other system of identifying location.</td>
<td>Critical for use in determining spatial correlation.</td>
</tr>
<tr>
<td>Constant Environment Characteristics</td>
<td>Describes the context in which the sensor is deployed.</td>
<td>Examples: soil type, liquid environment, or sensor packaging.</td>
</tr>
<tr>
<td>Physical Certainties</td>
<td>These features are based upon the natural laws of science.</td>
<td>Can be used to define a fault.</td>
</tr>
<tr>
<td>Environmental Perturbations</td>
<td>These are contextual features are not constant throughout the deployment lifetime.</td>
<td>Examples: Weather patterns, rain, or irrigation events.</td>
</tr>
<tr>
<td>Environmental Models</td>
<td>Models of the phenomenon behavior as defined by experts and computed from the data.</td>
<td>Examples: Expected rate of change or micro-climate models.</td>
</tr>
</tbody>
</table>

4.1 Environment Features

Environment features, or context, contribute greatly to models for expected behavior and fault behavior by describing the context in which a sensor is placed. Aside from sensor location, environmental features are mostly out of the control of the sensor network operator. These features are defined in Table I with examples of the feature or how the feature may be used.

Environmental features will always play a dominant role in the type and prevalence of faults because they play a significant role in determining expected behavior. This expected behavior in turn determines faults. All environmental features give context to a fault. Also, there may be uncertainty associated with some of these features, e.g., in the effects on the data by perturbations, or in the environmental models. Thus, we use confidence intervals to define the expected range of values.

4.1.1 Physical constants. These are constant factors that are not expected to change throughout the lifetime of the sensor deployment. This is made up of sensor location, constant environment characteristics, and physical certainties. As an example of a constant environment characteristic which may lead to faulty data, Szewczyk et al. [2004] suggests that since their sensor packaging was IR transparent, a mote would heat up in direct sunlight and report higher than expected temperatures.

Physical certainties are also known as the physical range in Mourad and Bertrand-Krajewski [2002]. An example of such a bound being exceeded is in Tolle et al. [2005] where the authors removed outliers that exceeded the physical possibility of 100% relative humidity.

4.1.2 Environmental perturbations. Environmental perturbations can be used to explain the causes of aberrant behavior. The effect of the environment has been noted to affect sensors in both Szewczyk et al. [2004] and Elmahrawy and Nath [2003]. For example, weather patterns and conditions may affect sensors in adverse ways. Rain can cause humidity sensors to get wet and create a path inside the sensor power terminals, giving abnormally large readings [Szewczyk et al. 2004].
Table II. Sensor Network System Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Definition</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transducer</td>
<td>The interface of sensor with the environment. Takes measurements of the phenomenon and produces a voltage output.</td>
<td>The reliability varies greatly dependent on sensor type. Transducer element is the component that is calibrated.</td>
</tr>
<tr>
<td>Analog-to-digital converter</td>
<td>Maps the analog voltage signal into a range of discrete values.</td>
<td>The ADC may limit the ability to detect features above or below the maximum or minimum ADC value.</td>
</tr>
<tr>
<td>Total detection range ($R_{\text{detection}}$)</td>
<td>Defines the transducer input-output curve regions. This is the overall range of values for which a sensor had been tested and calibrated.</td>
<td>Mapping may change over time due to sensor drift (section 5.2.1)</td>
</tr>
<tr>
<td>Interval of confident operation ($R_{\text{confident}}$)</td>
<td>The range of output values that can be confidently translated into input values.</td>
<td>Statistically determined bounds give a user defined confidence level.</td>
</tr>
<tr>
<td>Saturated interval ($R_{\text{saturated}}$)</td>
<td>Complimentary to $R_{\text{confident}}$. This range of output values cannot be reliably related to one input value with low uncertainty.</td>
<td>This range defines the environment out or range fault (section 5.2.4).</td>
</tr>
<tr>
<td>Sensor Age</td>
<td>Length of time sensor is deployed and actively monitoring.</td>
<td>Components can be expected to degrade over time.</td>
</tr>
<tr>
<td>Battery State</td>
<td>Amount of energy left in the battery relative to the minimum operating power required for sensor operation.</td>
<td>Low batteries can cause erratic, noisy, and/or unreliable measurements if any.</td>
</tr>
<tr>
<td>Noise</td>
<td>Random unwanted variation in data.</td>
<td>Causes uncertainty in data.</td>
</tr>
<tr>
<td>Sensor Response</td>
<td>Delay in sensor response to a change in the phenomenon.</td>
<td>Phenomenon may be captured incorrectly.</td>
</tr>
<tr>
<td>Sensor Network Modalities</td>
<td>Sensing tools that measure different but related phenomenon.</td>
<td>Can be leveraged to increase fault detection performance.</td>
</tr>
</tbody>
</table>

Environmental perturbations can also be leveraged in the modeling of expected behavior. For example, in Ramanathan et al. [2006], irrigation events that influence the concentration of chemical ions in soil are expected on a regular basis. By incorporating the prior knowledge of how the concentration should change due to irrigation, one can increase the accuracy of any type of model developed.

4.1.3 Environmental models. Environmental models are crucial in defining expected behavior of a phenomenon. The quality of the model in turn greatly affects performance of fault detection algorithms. For example, in the cold air drainage experiment described in Ni and Pottie [2007], temperatures are not expected to be homogeneous at the different sensor locations. If the authors had a model of the degree to which temperatures differed between each sensor, then it would have increased the fault detection ability.

4.2 System Features and Specifications

We now discuss features specific to individual sensors and features involving the overall sensor network; these may influence any model developed for behavior of the sensor network. We summarize the feature definitions and their significance as related to faults in table II.

ACM Journal Name, Vol. V, No. N, Month 20YY.
First, we examine features of individual sensors before moving to features of the sensor network which we can split into two general types. Sensor hardware features describe the components and abilities of a sensor, while calibration describes the uncertainty of the mapping from input to output.

4.2.1 Hardware components. Figure 1 is a diagram of a typical sensor and the flow of data through major components. These features describe the location where a fault may occur. Associated with each of the components are certain static limiting features, which may be defined by specifications, that may impact resulting data. However, as the sensor user does not have access to internal signals, we will only discuss the two most pertinent components: the transducer at the input and the analog-to-digital converter at the output.

An example of the transducer affecting sensor reliability is in ion selective electrode sensors. These sensors deployed in soil feature a chemically treated membrane that is not very robust and frequently fail in a deployment [Ramanathan et al. 2006]. As noted in table II, the analog-to-digital converter limits the detection ranges, and this plays a defining role in the clipping fault in section 5.2.5.

4.2.2 Calibration features. Calibration may be necessary to increase the accuracy of the sensor since factory calibration conditions may not always be relevant to conditions in the field. When referring to calibration, it is the transducer response that one calibrates, assuming there is no clipping by the analog-to-digital converter. Figure 2 is a general input-output calibration curve, similar to that of Ramanathan et al. [2006b] and [Rundle 2006]. Calibration features are used to describe faults.
The total detection range, $R_{\text{detection}}$, is comprised of the interval of confident operation, $R_{\text{confident}}$, and saturated interval, $R_{\text{saturated}}$. The $R_{\text{confident}}$ is usually linear and should consist of one-to-one mappings of output values to input values. Depending on the type of sensor, there may be different degrees of variability outside the interval of confident operation. The ISE chemical sensors exhibit a “flattening” in the data outside of $R_{\text{confident}}$ [Rundle 2006], while as we will see later, the ISUS nitrate sensor will exhibit higher output variance with larger input values.

4.2.3 Other system features. In addition to the sensor component specific features, there are higher level features of a sensor and sensor network that may be incorporated into a fault detection system model.

Sensor age can influence the reliability of a sensor. For example, the treated filtering membrane for a chemical sensor wears out over time giving faulty data. Similarly, battery life is seen to give unreliable measurements in Szewczyk et al. [2004], Ramanathan et al. [2006], and Sharma et al. [2007].

Noise can be modeled using a probability distribution, such as a Gaussian. While not always completely accurate, the Gaussian noise assumption is convenient to work with. An example of sensor response hysteresis affecting data quality is given in Bychkovskiy et al. [2003]. In an experiment measuring temperature of a heat source moving across a table, thermocouples have a slow response relative to the velocity of the heat source.

Different sensor network modalities can be leveraged to model sensor network behavior for fault detection. For example, humidity and temperature measurements should be correlated since the two affect one another. If the two do not correspond to the correlation model, then a fault has likely occurred. The use of different modalities has been mentioned in Szewczyk et al. [2004], Werner-Allen et al. [2006], and Buonadonna et al. [2005].

4.3 Data Features

Data features are usually statistical in nature. A confident diagnosis of any single fault may require more than one of these features to be modeled. We cannot provide a complete list of possible features and tools that can be used, but the included features are commonly exploited and simple to implement. These features are usually calculated in either the spatial or temporal domains. Data features are primarily used to describe or identify faults. We provide examples of where these features have been used in literature, and table III summarizes the usage of these features.

As discussed previously, features are commonly calculated or modeled over a window of samples. Windowing may be done over the temporal domain or over space by selecting sensors that are expected to retain similar characteristics, usually colocated or nearby sensors.

The mean and variance are commonly exploited basic statistical measures. Means and variances can be calculated in a moving average context, as in Mourad and Bertrand-Krajewski [2002] for data smoothing. Jeffery et al. [2006] uses the mean across both temporal and spatial windows to correct for faulty sensor values. The variance or standard deviation is also a measure of the reliability of a sensor, since high variance is often a sign of faulty data. [Sharma et al. 2007].
Table III. Sensor Network Data Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Usage in fault modeling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean and Variance</td>
<td>The mean and variance can be used to determine expected behavior via regression models or correcting faulty sensor values. Variance also is a measure of reliability.</td>
</tr>
<tr>
<td>Correlation</td>
<td>Assumed or stated spatial or temporal correlation models are required for regression methods to have meaningful use.</td>
</tr>
<tr>
<td>Gradient</td>
<td>Rate of change on different scales, e.g. over 10 minutes or 24 hours etc., can be used in modeling faults.</td>
</tr>
<tr>
<td>Distance from other readings</td>
<td>Distance between data is used to directly or indirectly to determine if data is faulty.</td>
</tr>
</tbody>
</table>

In a sensor network, data is expected to be correlated in both the spatial domain and temporal domain for sensor networks. This correlation can be vaguely defined as in Ni and Pottie [2007], Jeffery et al. [2006], and firmly defined probabilistically as in Elnahrawy and Nath [2004]. Spatially, Balzano and Nowak [2007], as well as the previously mentioned works, seek to exploit correlation models to improve sensor network performance.

The gradient is exploited in Sharma et al. [2007] and Ramanathan et al. [2006] for fault identification. The scale selection over which the gradient is calculated is a nontrivial task and will depend on the type of phenomenon being observed. If the phenomenon is slow moving, such as temperature, the scale may be longer than a highly varying phenomenon, such as wind velocity.

To use the distance from other readings indirectly, one would compare data with a model of expected behavior, which may be as simple as the mean from nearby sensors or recent data values as in Jeffery et al. [2006]. To use such a feature, one may use static thresholds as in Ramanathan et al. [2006] or thresholds based upon an estimated probability distribution and confidence level as in Ni and Pottie [2007].

There are many more statistical techniques, spatio-temporal and otherwise, that have been used to model sensor data, but not in the context of fault detection. Gaussian processes have been used to model the environment for sensor placement [Krause et al. 2006]. Additionally, other methods such as Kriging and variograms may prove useful in future works.

Data features are commonly employed to identify faults in fault detection algorithms. It is useful to combine data features for detection of certain faults. For example, in section 5.1.1 variance and gradient are given as examples for detection of outliers. As mentioned previously, the time scale over which a fault detection method evaluates data plays a critical role in determining how useful a particular data feature is effective. Also, the usefulness of a particular feature is dependent on the type of fault. The most useful features for each fault are summarized in table V.

5. FAULTS

With the feature list in place, we now define the most common faults observed in a sensor network. Faults carry different meanings as to their ultimate interpretation and importance. Depending on the context and sensor network application, some
Table IV. Relating system view and data view manifestations.

<table>
<thead>
<tr>
<th>Data-centric fault</th>
<th>System view fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outlier</td>
<td>Connection/Hardware</td>
</tr>
<tr>
<td></td>
<td>Low Battery</td>
</tr>
<tr>
<td>Spike</td>
<td>Clipping</td>
</tr>
<tr>
<td></td>
<td>Connection/Hardware</td>
</tr>
<tr>
<td></td>
<td>Low Battery</td>
</tr>
<tr>
<td>Stuck-at</td>
<td>Low Battery</td>
</tr>
<tr>
<td></td>
<td>Connection/Hardware</td>
</tr>
<tr>
<td>Noise</td>
<td>Environment out of range</td>
</tr>
<tr>
<td></td>
<td>Calibration</td>
</tr>
</tbody>
</table>

faults will still have informational value, while others are totally uninterpretable and the data must be discarded. We will point out examples of this grey scale interpretation of faults and summarize the impact in table VI.

Unless ground truth is known or given by something with high confidence, the term fault can only refer to a deviation from the expected model of the phenomenon. When defining a fault, there are two equally important approaches, and it may be easier to describe a fault using one approach over the other. Frequently there may not be a clear explanation as to the cause of a fault, e.g. outliers, and hence it may be easier to describe this fault by the characteristics of the data behavior. This is the “data-centric” view for classifying a fault and can be seen as a diagnostic approach. The second method, a “system view,” is to define a physical malfunction, condition, or fault with a sensor and describe what type of features this will exhibit in the data it produces.

These two approaches are not disjoint and can overlap. A fault defined using one approach can usually be mapped, as depicted in Table IV, into one fault or a combination of faults defined using the other approach, and vice versa.

For each fault, we will provide examples and discuss the features that are most relevant for modeling the faults, and provide examples of how to model each fault. Where appropriate we will discuss the effect of the time scale and how human feedback can improve modeling for systems, thus reducing system supervision. Also when possible we will discuss the overlap between the system and data-centric views. In certain cases we discuss the interpretation and importance of the fault in question. As we do not seek to design or promote any particular fault detection algorithm, we only present very simple illustrative examples of how such fault models may prove useful in practice.

5.1 Data-centric view

We first examine faults from a data-centric view where we determine a fault based upon data from a sensor. Data-centric faults may or may not be reproducible depending on the cause.

5.1.1 Outliers. Outliers are one of the most commonly seen faults in sensor data. We define an outlier to be an isolated sample, in the temporal sense, or a sensor, in the spatial sense, that significantly deviates from the expected temporal or spatial models of the data which are based upon all other observations. The
temporal version of an outlier has been classified as a SHORT fault and subjectively described in Ramanathan et al. [2006] where the fault is subjectively described. Outlier detection is not new, as this issue has existed for a long time [Hodge and Austin 2004]. More recently Sheng et al. [2007] has the primary focus of outlier detection in sensor networks.

We provide an example in figure 3 where there are clear outliers in the data. This example only considers temporal outliers, but methods described here can easily be translated to spatial outliers such as in figure 6(a). Figure 3 is humidity data in the form of raw output of the sensor (which can be converted to relative humidity percentage) [Kaiser et al. 2003] [NIMS 2007]. Two of these outliers have been inserted by software declaring a communication issue (indicated by a −888) or a data-logger problem (indicated by a −999). While some of these outliers have known causes, many other outliers are completely unexpected.

![Figure 3](image_url)

**Fig. 3.** Raw humidity readings from a NIMS deployment with examples of outliers

To model an outlier, the most common features to consider are distance from other readings as in Sheng et al. [2007] and gradient as in Ramanathan et al. [2006]. As defined, we must first model the underlying expected behavior. For the purposes of demonstration we will use simple methods of determining the expected range based upon previous data sample points.

Contextual information about the phenomenon and sensor plays a larger role in this fault since we are modeling expected behavior in an effort to identify outliers. As this is humidity data, we assume, based on an environmental model assumption, that this phenomenon does not change rapidly. One can better define this environmental model in a mathematical form to describe expected rate of change or other aspects of the data, however this is beyond the scope of this paper. This model assumption is the basis for the window size selection and how we model the expected behavior.

We pick one particular outlier, at (2.044, 8.469), to examine and model the first feature of distance. We define features to be based upon the previous 75 samples, or approximately half an hour as we do not expect humidity to change much over this time. A homogeneous data assumption allows us define the expected model using the sample mean and variance features. We define the 95% confidence interval to be the expected range, and anything outside of this can be considered an outlier. Other possible modeling measures include the median and quartiles, regression models, and other more complex models.

After modeling the previous half hour before the sample point in question, we determine a sample mean of 1817.7 and a standard deviation of 14.923 resulting
in a confidence interval of $[1787.9, 1847.6]$. The sample point being considered is compared to this, and anything outside this expected range will be marked as an outlier, e.g. $(2.044, 8.469)$.

One can apply similar techniques for developing a confidence interval for the gradient feature. By modeling the mean absolute point-to-point change or using a first order linear regression to estimate the gradient one can construct a confidence interval for the gradient and identify outliers.

The selection of the window here affects the accuracy of the very basic models developed for the expected behavior. The homogeneous assumption is less accurate as the window size is increased. On the other hand, using smaller window sizes can lower the accuracy of the model as less data is used for modeling. Developing more complex models of the expected behavior diminishes the effect of window size for outlier detection.

Outliers are most commonly not very informative, and hence can usually be discarded, e.g. Tolle et al. [2005]. The effect of keeping the outlier in the data set can significantly alter the model and allow for more missed detections since any new model may be based upon faulty data.

5.1.2 Spikes. We define a spike to be a rate of change much greater than expected over a short period of time which may or may not return to normal afterwards. It is a combination of at least a few data samples and not one isolated data reading as is the case for outliers. It may or may not track the expected behavior of the phenomenon. While it may not always be a fault, it is anomalous behavior and thus should be flagged for further investigation.

As Mourad and Bertrand-Krajewski [2002] suggests, determination of spikes must be based on environmental context and models of the physical phenomenon. For example, light data in figure 12 can experience sudden and large changes in gradient, however in this context, this cannot always be judged to be a fault since light is a phenomenon that can give large gradients. By contrast, in the example that follows in this section, a spike is not expected to occur in this soil concentration application.

Good models, improved by human knowledge, of the phenomenon will allow for proper distinction in cases of uncertainty. Similarly, context and environmental models will dictate the time scale judged to be “a short period of time.”

We look at an example from a deployment in Bangladesh as described in Ramanathan et al. [2006] and Ramanathan et al. [2006b]. Figure 4 is the concentration of ammonium reported at one sensor location. There are two examples that we define as spikes. The first example occurs at the time frame 1.3805 to 1.3875 days over the four data samples and returns to normal behavior. The second example occurs between 7.8607 to 9.511 days and persists for a while.

By definition, the primary feature for modeling is temporal gradient. Other data features that may be useful for detection are mean and temporal correlation. We will first discuss temporal gradient.

One method of modeling a spike is to determine an expected range for the gradient by modeling local rate of change across a window size using a regression or another model. Then, one can construct a confidence interval about this range similar to that of the outlier case. A spike can then be modeled as having a gradient larger
For an example, we use the first spike between time frame 1.3805 to 1.3875 days. We use a first order regression to model the expected gradient of the log of the data to be 7.0518 using data from the previous hour. The beginning of the spike itself has a gradient of $-729.11$, which is outside of any reasonable confidence interval around the expected gradient. Hence, if one were to generate a model for a spike, one would create a set of data that has a gradient much higher than expected.

Next, we look at temporal correlation. If one has an assumption that sensor values are expected to be correlated to some degree, as is usually the case in sensor data, then this feature may prove useful. That is, if we expect some linear correlation, one can calculate the correlation for the data. In the data prior to the 7.8607 spike, the correlation coefficient is 0.46536. However, as we expect, there is a drastic drop in correlation to $-0.44338$ once the spike is introduced into the data which signals that there is an anomaly.

Additionally, Mourad and Bertrand-Krajewski [2002] uses the mean data feature and calculates a moving average to smooth data. A spike would then be defined by having the residue between the data and the moving average exceeding a defined threshold or confidence interval.

5.1.3 “Stuck-at” fault. A “stuck-at” fault is defined as a series of data values that experiences zero or almost zero variation for a period of time greater than expected. The zero variation must also be counter to the expected behavior of the phenomenon. The sensor may or may not return to normal operating behavior after the fault. It may follow either an unexpected jump or unexpected rate of change. The data around such a fault must exhibit some variation or noise for one to detect this fault since variation is the distinguishing characteristic of the fault.

While similar to the “CONSTANT” fault in Sharma et al. [2007] and Ramanathan et al. [2006b], we differ in that the value in which the sensor may be stuck may be within or outside the range of expected values. In cases where the stuck at value is within the expected range, spatial correlation can be leveraged to identify whether the stuck sensor is faulty or functioning appropriately.

By definition, the primary feature to consider modeling is variance. Spatial correlation can also be considered especially when the stuck-at fault occurs inside the range of expected values for the phenomenon.

Human input can initially define the length of time, i.e. time scale, for which a sensor is stuck before it is considered to be a “stuck at” fault based on the sensing
context and the expected variability of the readings. If little variability is expected, there should be greater tolerance for having very little variation for a greater period of time. With further development, this can be incorporated into a model and the human involvement can be reduced.

Figure 5 shows the chlorophyll concentrations from two buoys in a NAMOS [2006] deployment at Lake Fulmor monitoring the marine environment. This data exhibits little variation after two unexpected changes in gradient. The flat tops of the chlorophyll concentration for node 103 in figure 5(b) indicate that there has likely been a stuck at fault. Furthermore, there is little or no variation at those samples near that value. For example, we calculate the variance of the data in a half day window preceding the first “stuck-at” instance to be approximately 4712. The variance of the entire first “stuck-at” instance is approximately 1.7. While this is not 0, further analysis of the data shows that there are large pockets of time during this instance where the variance is zero, and the only variation is that the sensor values vary only slightly outside these pockets. Similarly, for sensor node 107 in figure 5(a), the period between days 0.74 and 0.974 has an overall variance of 0.000007, with large pockets of 0 variance. The data prior to this period has an overall variance of 85.3.

To ensure that only the sensor is behaving in such a manner and it is not the phenomenon, especially in cases where precision is low and the fault occurs within the expected range, spatial correlation can be leveraged to increase confidence in detection. If there is a model indicating there should be correlation between two sensor locations for the data or variances, then spatial correlation can be used to determine whether or not a “stuck-at” fault is actually a fault. In figure 5(a) we can see that sensor 102 does not exhibit the fault that sensor 107 has. Since two sensors are expected to be correlated, which is the case for times outside of the fault, we can reasonably conclude that sensor 107’s data is faulty.

Alternatively, light data in figure 12 exhibits a “stuck-at” fault within the expected range of the phenomenon indicating clipping. However, high spatial correlation among the sensors suggests that this is normal behavior, and the sensor is not actually malfunctioning.

Data from a “stuck-at” fault may not always be thrown away as sometimes the data may still provide some information concerning the phenomenon. In the case of sensor clipping in the light data, the “stuck-at” fault still identifies that the light is at least greater than or equal to the reported value. However in the NAMOS data presented in this section, data may be discarded as there is no useful interpretation.
5.1.4 High noise or variance. While noise is common and expected in sensor data, an unusually high amount of noise may be a sign of a sensor problem. Unusually high noise may be due to a hardware failure or low batteries, as in sections 5.2.2 and 5.2.3. We define a noise fault to be sensor data exhibiting an unexpectedly high amount of variation. The data may or may not track the overall trends of the expected behavior. This fault is also presented in Sharma et al. [2007], but we emphasize that the noise must be beyond the expected variation of the phenomenon and sensor data.

As defined, the primary feature of interest is the variance. Spatial correlation of data and/or the moments of the data also may also be useful in judging the nature of the fault.

We provide examples of a high noise fault from data collected from the cold air drainage deployment. In figure 6(a) we plot data from three nearby sensors from a cold air drainage deployment in March 2006 [CAD 2006-2007]. One sensor is clearly faulty and has a considerable amount of noise in addition to the spatial outlier behavior. Also in the cold air drainage data of figure 6(b), there is one sensor that has a high amount of noise, yet still tracks the data. Unlike figure 6(a), this data tracks the expected behavior of the phenomenon.

![Fig. 6. Cold air drainage temperature data from two different time periods.](image)

The initial selection of proper window size within which to model data and calculate the variance is dependent on modeling assumptions. If modeling similar to the regression model in Ni and Pottie [2007] were used to estimate the variance around the expected value, then a larger window size may prove to be more accurate in estimating the sensor variance. However, if sensor variance is directly computed, a large window may produce an artificially high variance due to natural variations in the phenomenon, e.g., diurnal patterns in the figure 6(b).

The expected variance of the sensor readings is given by either a data sheet of the sensor, model of other similar sensors, environmental understanding, or past behavior of the sensor in question. If the environment is expected to have a high variance, then this may not be considered a fault.

In the case of the cold air drainage data, an expected range for the variance can be based upon the other sensors. The variance of a sensor experiencing a noise fault should exceeds this expected behavior.

Correlation across sensors of the moments (2nd or higher) of the data or variance features may also be leveraged to increase detection for data expected to have high variability. If a particular sensor has unexpected variability, and another nearby sensor also has high variability on the same scale, then it is less likely that a fault...
occurred. However, if there is no correlation in variability then it is more likely that a fault has occurred.

Examining the data in figure 6(a), we perform a rough estimate of the variance over an approximately 2 hour moving window. With this we examine the correlation, or covariance structure of the variance. Taking the overall pairwise linear correlation between each sensor, we find that the correlation between the two reliable sensors’ variances is 0.6 indicating that these two sensors are correlated. The pairwise correlations of the variances between the reliable sensors and the faulty sensors are 0.1 and 0.2 indicating that the faulty sensor variance is not similar to the other sensors’ variances.

Noisy data may still provide information regarding the phenomenon at a lower confidence level. Therefore, if the noisy data tracks the expected behavior as is the case in figure 6(b), then it should not necessarily be discarded. Data from figure 6(a) may be discarded as this is also a case of a spatial outlier.

5.2 System-centric view
We give general reasons for a sensor failure and detail how a sensor might behave with a certain fault with examples from real world deployments. Also, monitoring certain aspects of the hardware, such as battery life, may aid in understanding when a fault may occur.

5.2.1 Calibration Fault. Calibration problems can be a root cause of faulty data in many cases. Many papers cite the difficulty in calibration, especially while the sensor network is deployed [Buonadonna et al. 2005] [Bychkovskiy et al. 2003] [Balzano and Nowak 2007] [Ramanathan et al. 2006b]. The result of calibration errors is that one gets a lower accuracy of sensor measurements but not necessarily lower precision. We discuss three different types of calibration errors which are named in the previously cited works:

—Offset fault - Sensor data values are offset from the true phenomenon by a constant amount. The data still exhibits normal patterns over an extended period of time.

—Gain fault - The rate of change of the measured data does not match with expectations over an extended period of time. That is, whenever the phenomenon changes by any amount, $\Delta$, then the sensor reports a change of $G \times \Delta$, where $G$ is a positive real value.

—Drift Fault - Throughout a deployment of a sensor, sometimes performance may drift away from the original calibration formulas. That is, the offset or gain parameters may change over time.

Because these errors may be combined in several ways, calibration errors can manifest themselves in many different ways. In many cases, this makes detection and modeling of general calibration errors difficult without human input or ground truth. Even with human input, when lacking ground truth it may be difficult to differentiate between mis-calibration and natural phenomenon variations.

While calibration errors are defined relative to ground truth, without ground truth, calibration faults can only be determined relative to an expected model. This model can be a predefined model, a model generated from correlated sensors,
or a combination of both. The predefined model is based upon environmental context which may include micro-climate models. Spatial correlation is important for generating the expected model when lacking ground truth, as is exploited in Bychkovskiy et al. [2003] and to some extent Balzano and Nowak [2007].

We present an example of what can be considered a calibration fault in figure 7 presented in Ramanathan et al. [2006b]. One of three sensors monitoring carbon dioxide concentration at various levels within the soil exhibits unusual sensor readings when compared to the other sensors.

![Graph](image)

Fig. 7. CO$_2$ soil concentration at three different depths at a deployment in James Reserve. The sensor at 16cm has some calibration issues.

The sensor at 16cm is clearly not measuring what is expected for the majority of the time, however while accuracy has changed, precision has not. There are also some similarities with the other sensors and exhibits. For example there is a common spike in all three sensors prior to day 200. There is also a drift fault, since the offset changes with respect to time. Also, eventually the sensor returns to normal operation.

The data of figure 12 in section 5.2.5, presents another example of calibration error. While not as serious as the previous example, the lowest value of the “floors” differ when it is expected they report the same values. Hence, this slight difference is a sign of what is likely an offset fault.

Faulty data due to calibration issues still provides useful insight about the phenomenon and should not be readily discarded. If a proper calibration formula were to be developed, it is possible that the data may be corrected with acceptable confidence.

5.2.2 Connection or hardware failures. Frequently sensors may fail due to hardware problems such as poor connections. This is a general feature category since it is not possible to characterize all possible sensor failure modes. Typically, hardware failures require either replacement or repair of a sensor. This is one of the more common issues that may arise in a sensor deployment and has been cited as a cause of sensor failures in Szewczyk et al. [2004], Ramanathan et al. [2006], and Sharma et al. [2007].

A connection or hardware fault will often manifest itself by reporting unusually high or unusually low sensor readings. These readings can even occur outside of the feasible environmental range. For example humidity outliers were discarded in when the relative humidity exceeded physical possibilities in Tolle et al. [2005].

One cause of hardware faults is weather or environment conditions. Szewczyk et al. [2004] cite water contact with temperature and humidity sensors causing a
short circuit path between the power terminals as the cause for abnormally large or small readings. Including weather conditions in a model for the probability of failure can increase the likelihood of fault detection when an environmental event occurs, e.g. rain.

In another NAMOS deployment, thermistors failed due to prolonged exposure to water which created a bad connection within the sensor. Two sensors are giving anomalous and likely faulty data in figure 8. These faults are due to bad connections in the thermistors. There are several other sensors recording reasonable values, but the two faulty sensors are clearly out of any reasonable range as defined by an environmental and data model.

The fault behavior from connection or hardware faults are often sensor dependent. The datalogger may have software to choose to report certain values if something is out of the range of the data logger. For example, for the NIMS humidity data set in section 5.1.1, the datalogger will report a −999 when it receives a value out of the range. However, in the NAMOS example given here, the sensors will continue to sample even as the values seem to be clipped by the range of either the thermistor or the ADC.

Hardware may also fail in other ways beyond electrical malfunctions. For example, the ion-selective electrode sensors used in soil deployments are often prone to failures [Ramanathan et al. 2006b]. A chemically treated membrane filtering the ion of interest in the sensor is prone to failure when deployed in the field. There may also be interference from other ions present that cause data to be inaccurate [Rundle 2006].

Human interaction plays a very important role when diagnosing an unknown hardware issue. Since it is not possible to detail every way a sensor can fail, a person’s ability to investigate and provide an explanation for a fault is invaluable. Once a fault is diagnosed, its behavior recorded and incorporated in a future automated expert diagnosis tool, the future role of a person is reduced.

Once a hardware fault is detected, then it may be best to discard the data. Since the sensor is not performing as it was designed, the data it reports is likely not usable.

5.2.3 Low battery. Another reason for faulty or noisy data is a low battery voltage, a primary feature of the system as stated in section 4.2.3. Battery life is an important measure of sensor health [Szewczyk et al. 2004] [Ramanathan et al. 2006b] [Tolle et al. 2005]. Low battery levels are not only an indication of how long
a sensor will last as it can also influence sensor readings in various ways and cause less reliable or faulty data.

An example provided in Ramanathan et al. [2006] illustrates one possible outcome of a low battery; readings from a sensor with low batteries may experience a noise fault, as in section 5.1.4. When a weak battery is replaced, the variance of the data samples dramatically decreases by more than threefold. Also in the cold air drainage data in figure 6(b), it is likely that the noisy sensor is due to a low battery. While the data still tracks the expected behavior, the noise is much greater than expected.

Another way a battery may affect sensor data samples, is that sensors may begin to report unreasonable readings. There may be an unexpected change in gradient as in the following example. At another NAMOS deployment, one buoy’s battery was old and hence it did not have as much capacity. In figure 9 there is a drop in temperature in the last hours before the sensors stopped reporting, which is a spike fault as described in 5.1.2. We can also see that sensors may also exhibit a “stuck-at” fault following a spike when the battery level falls too much. From the Intel Lab at Berkeley data set [Intel 2004], we plot in figure 10 two nearby motes’ reported temperature values and the battery voltage. Both sensors begin to fail at approximately the same voltages indicating that failure is a likely due to insufficient power. Once battery voltages drop below this value, the temperature

Fig. 9. Readings from three thermistors at buoy 112 for an August 2007 deployment of NAMOS sensors in Lake Fulmor. Sensors values drop significantly as batteries fail, other thermistors behave similarly.

Fig. 10. Temperature readings and battery voltages from two nearby motes in the Intel-Berkeley Lab data. The horizontal line provides an approximate voltage level at which both sensors begin to fail.
sensors exhibit a spike, with excessive gradient, and then remain “stuck-at” one particular value for the rest of the deployment. This is also exemplified in Tolle et al. [2005] where sensors’ battery failures correlated with most of the outliers in the data. When the battery voltage level was less than 2.4V or greater than 3V, behavior similar to that of figure 10 manifested itself.

Battery supply can affect system performance significantly by either adding noise or giving faulty data depending on the type of sensor. In some cases, it may be worthwhile to keep the data, as in figure 6(b), as the data still retains information about the phenomenon. More often, data might be uninterpretable and must be discarded as is the case for the data in figures 10 and 9.

5.2.4 Environment out of range. There may be cases in which the environment lies outside of the sensitivity range of the transducer. The manifestation of this issue influenced by the calibration feature of the sensor was discussed in section 4.2.2.

We present two examples of common behavior when the environment is out of the range of the transducer. In the deployment of chemical sensors mentioned in Ramanathan et al. [2006], one chloride sensor reported concentrations outside of the total detection range, figure 11(a). The entire range for which the sensor was measured to have sensitivity is denoted by the horizontal lines. At the extremes, the data experiences a flattening. The sensor readings end up being predominantly outside of this range. While there are still some slight diurnal patterns, the values remain outside of the measured sensitivity range, and hence there is little confidence in these data values.

5.2.5 Clipping. Clipping is exhibited when a sensor seems to have maxed out, and is usually caused by the environment exceeding the limits of the analog to digital converter, $R_{ADC}$. This type of error mentioned in the context of light sensors in Szewczyk et al. [2004] where the sensors saturated at the maximum ADC value and 0. While this is not exactly a sensor fault as the sensor is only operating within
its designed parameters, there is reduced confidence for the data when the sensor reaches its maximum.

This fault usually manifests itself as a “stuck-at” fault for consecutive values at the extremes of the data range. Hence, the important features for detection and modeling to examine are the same as described in section 5.1.3. Also, this fault may follow a sudden change in gradient at the extreme values of the data range.

As we will see, by considering the context and values of the fault, one can identify the “stuck-at” fault to be clipping. Figure 12 shows light data from two motes along the same wall at the Intel Lab at Berkeley deployment.

\begin{figure}[h]
\centering
\subfigure[Node 24]{
\includegraphics[width=0.4\textwidth]{node24.png}}
\subfigure[Node 32]{
\includegraphics[width=0.4\textwidth]{node32.png}}
\caption{Data from two of the light sensors deployed at the Intel Research Berkeley lab.}
\end{figure}

In the middle of each day, the maximum value at which the sensor peaks is 1847.36 for both sensors and does not move beyond. Any variation in the data does not exceed this value. Following a sudden change in gradient to 0, this data behaves as a “stuck-at” fault. Since these two sensors, as well as other collocated sensors, behave similarly, then by taking advantage of an expectation of spatial correlation, one can reasonably conclude that the cause is clipping. Thus the environment in this case has exceeded either the $R_{\text{detection}}$ or $R_{\text{ADC}}$.

The inference that the environment exceeded the upper limit of the sensor is based upon the underlying environmental assumptions made. We have made the reasonable assumptions that light in the lab exceeds 1847.36 and that there should be variations in the light scale during the time of clipping.

These assumptions on the environmental model and spatial correlation may change depending on the context affecting the detection of this fault. For example, examining the lowest light values of figure 12(a), the values are not consistently the same, so it is more difficult to conclude clipping occurred. Additionally, there is a lower bound for light intensity, so light may not actually drop below measurable limits. Spatial correlation also does not provide a clear cut conclusion either. The data from node 32 does not have the same minimum values as node 24, likely due to a slight calibration error; this adds to uncertainty in a conclusion of clipping.

As mentioned in section 5.1.3, clipped data may still provide reduced informational value for interpretation by the scientists. Hence, data exhibiting such behavior should not be discarded.

5.3 Confounding factors

There may also be confounding factors that influence sensor readings. For example temperature may influence chemical sensors, there may be interfering ions in the chemical sensors. As in figure 10, we see that the battery level actually fluctuates...
Table V. Taxonomy of faults: Definitions and possible causes.

<table>
<thead>
<tr>
<th>Fault</th>
<th>Definition</th>
<th>Indications and Possible Causes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outlier</td>
<td>Isolated data point or sensor unexpectedly distant from models.</td>
<td>The distance from other readings is beyond expectations. The gradient changes greatly when the outlier is included. Causes are often unknown unless software inserted by datalogger.</td>
</tr>
<tr>
<td>Spike</td>
<td>Multiple data points with a much greater than expected rate of change.</td>
<td>A sudden change in gradient which is greater than expected. Little temporal correlation between historical data and the spike. Frequent causes include battery failure and other hardware or connection failures.</td>
</tr>
<tr>
<td>“Stuck-at”</td>
<td>Sensor values experience zero variation for an unexpected length of time.</td>
<td>Variance is close to zero or zero. Spatial correlation can be leveraged to determine whether or not in-range stuck-at values are faults. Frequently the cause of this fault is a sensor hardware malfunction.</td>
</tr>
<tr>
<td>High Noise or Variance</td>
<td>Sensor values experience unexpectedly high variation or noise</td>
<td>Variance is higher than expected or historical models suggest. Spatial correlation can be used to judge whether or not variation is due to the environment. This may be due to a hardware failure, environment out of range, or a weakening in battery supply.</td>
</tr>
<tr>
<td>Calibration</td>
<td>Sensor reports values that are offset from the ground truth.</td>
<td>Calibration error and sensor drift is the primary cause of this fault. A sensor may be offset or have a different gain from the truth. The amount of each may drift with time.</td>
</tr>
<tr>
<td>Connection or Hardware</td>
<td>A malfunction in the sensor hardware which causes inaccurate data reporting</td>
<td>Behavior is hardware dependent. Common features include unusually low or high data values, frequently exceeding expected range. Environmental perturbations and sensor age may indicate higher probabilities of failure. Other causes include a short circuit or a loose wire connection.</td>
</tr>
<tr>
<td>Low Battery</td>
<td>Battery voltage drops to the point where the sensor can no longer confidently report data.</td>
<td>Battery state is an indicator for system performance. Common behaviors include an unexpected gradient followed by either lack of data, or zero variance. There may also be excessive noise.</td>
</tr>
<tr>
<td>Environment out of Range</td>
<td>The environment exceeds the sensitivity range of the transducer.</td>
<td>There may be much higher noise or a flattening of the data. It may also be a sign of improper calibration.</td>
</tr>
<tr>
<td>Clipping</td>
<td>The sensor maxes out at the limits of the ADC</td>
<td>The data exhibits a “ceiling” or a “floor” at the data extremes. This is due to the environment exceeding the range of the ADC.</td>
</tr>
</tbody>
</table>
Table VI. Duration and Impact of faults

<table>
<thead>
<tr>
<th>Fault</th>
<th>Duration</th>
<th>Impact of fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outlier</td>
<td>Randomly occurring and instantaneous.</td>
<td>It may significantly skew the mean, variance, gradient, and other data features if not detected. It does not offer any useful information value and can be discarded.</td>
</tr>
<tr>
<td>Spike</td>
<td>Fault consists of more than one point. May or may not be temporary.</td>
<td>Spikes generally do not hold any informational value and should be discarded. This results in a loss of sensor data yield.</td>
</tr>
<tr>
<td>“Stuck-at”</td>
<td>Fault consists of more than one point. May or may not be temporary.</td>
<td>If cause is environment out of range or clipping, still holds informational value and data can be interpreted at lower fidelity. Otherwise data can be discarded.</td>
</tr>
<tr>
<td>High Noise or Variance</td>
<td>Fault consists of more than one point. Usually not temporary.</td>
<td>If the noisy data tracks other sensors, then the data still offers value and should not be discarded.</td>
</tr>
<tr>
<td>Calibration</td>
<td>Fault remains throughout the deployment.</td>
<td>Data should not be discarded. Uncalibrated data can still provide insight. A proper calibration formula can correct the data.</td>
</tr>
<tr>
<td>Connection or Hardware</td>
<td>Permanent once it occurs.</td>
<td>Data is meaningless as sensor is not performing as designed. Should be discarded.</td>
</tr>
<tr>
<td>Low Battery</td>
<td>Permanent until battery replacement or recharge.</td>
<td>Commonly, a battery failure results in useless data which should be discarded. The exception is if sensor behavior at low voltage gives added noise, then there may still be informational value.</td>
</tr>
<tr>
<td>Environment out of Range</td>
<td>Sensor returns to normal operation after environment returns to within range.</td>
<td>Still holds some information content. At minimum, indicates environment exceeds the sensor sensitivity range.</td>
</tr>
<tr>
<td>Clipping</td>
<td>Sensor returns to normal operation after environment returns to within range.</td>
<td>Still holds some information content. Indicates data exceeds the upper or lower ADC values.</td>
</tr>
</tbody>
</table>

with respect to temperature. The result of this is that these factors may influence sensor behavior and the fault likelihood.

Sensor faults can have multiple contributing factors, and other sensing modalities within the network may be leveraged to detect faults. For example, temperature and humidity are usually well related and can be combined to detect faults. As suggested earlier in section 4.2.3, one may incorporate relevant modality features when modeling data or faults.

Also multiple faults may occur at the same time, for example, a battery fault can cause a spike and a stuck at fault at the same time. A falling battery voltage will also cause calibration issues and cause the sensor to drift.

Finally, table V gives an overview of the faults their relevant features. While not specifically stated, environmental context plays a role in each one of the faults to determine the expected behavior of the sensor data. Table V also provides possible
causes that lead to particular faults. Table VI summarizes the practical impact and duration of these faults.

6. CONCLUDING REMARKS

We have provided a list of features which are commonly used for modeling sensor data and sensor data faults. With this, we provided a list of commonly exhibited sensor data faults which one can then use to test a specific fault detection system. There are many interactions between features and faults which make fault detection so difficult. However, we have presented a systematic way of looking at sensor data faults which could ease the next step of fault detection.

With this understanding of many possible faults, one can then develop more context-specific diagnosis systems. The next step would be to use an expert system [Jackson 1998] for a rules-based diagnosis system. Given data and faults behaviors, causes for faults are determined based upon these rules. Expanding this expert system into a Bayesian network [Heckerman 1995], a system would assess probabilities for the causes of faults, giving a likelihood that a certain data fault was caused by particular failure.

7. ACKNOWLEDGEMENTS

We would like to thank Tom Harmon, Robert Gilbert, Henry Pai, Tom Schoellhammer, Eric Graham, Gaurav Sukhatme, Bin Zhang, and Abishek Sharma for helping with the data collection.

This material is based upon work supported by the NSF under award #CNS-0520006. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

REFERENCES


ACM Journal Name, Vol. V, No. N, Month 20YY.


Received Month Year; revised Month Year; accepted Month Year