Suelo: Human-assisted Sensing for Exploratory Soil Monitoring Studies

Nithya Ramanathan
Center for Embedded Networked Sensing
nithya@cs.ucla.edu

Thomas Schoellhammer
Cisco / Ironport Systems
tschoell@cs.ucla.edu

Eddie Kohler
UC Los Angeles / Meraki
kohler@cs.ucla.edu

Kamin Whitehouse
University of Virginia
whitehouse@virginia.edu

Thomas Harmon
UC Merced
tharmon@ucmerced.edu

Deborah Estrin
Center for Embedded Networked Sensing
destrin@cs.ucla.edu

Abstract

Soil contains vast ecosystems that play a key role in the Earth's water and nutrient cycles, but scientists cannot currently collect the high-resolution data required to fully understand them. In this paper, we present *Suelo*, an embedded networked sensing system designed for soil monitoring. An important challenge for Suelo is that many soil sensors are inherently fragile and often produce invalid or uncalibrated data. Therefore Suelo is an *assisted sensing system*: it actively requests the help of a human when necessary to validate, calibrate, repair, or replace sensors. This approach allows us to use available sensors without sacrificing data integrity, while minimizing the human resources required. We tested our system in multiple real soil monitoring deployments and demonstrate that, using human assistance, Suelo produced 91% fewer false negatives and false positives than common fault detection solutions on these datasets.

Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and Embedded Systems

General Terms

Reliability

Keywords

Fault Detection, Fault Diagnosis, Soil Monitoring

1 Introduction

Throughout history, few things have mattered more to human civilization than soil, and nearly every plant and animal depends on soil ecosystems for nutrients, food, and water. However, soil ecosystems are complex, elusive, and still not well understood, and are considered by many to be the "final frontier" in science [12,20]. Today a scientific understanding of soil is more important than ever as agricultural and industrial practices have an increasing effect on the Earth's water and nutrient cycles. For example, tens of millions of people in the Ganges Delta continue to drink groundwater that is dangerously contaminated with arsenic, in what is perhaps the largest environmental poisoning in history. If consumption of contaminated water continues, the prevalence of arsenicosis and skin cancer in Bangladesh alone is estimated to reach 2,000,000 and 100,000 cases *per year* respectively, with deaths reaching 3,000 cases per year [39].

The current state-of-the-art measurement technique is called an *observation well*, which is a slotted tube inserted into the ground that allows water samples to be brought to the surface using pumps. These samples are then transported to laboratories or analyzed using a field kit. Observation wells require substantial human effort for each data reading, and their spatio-temporal sensing resolution is therefore limited; a single person who must read from 25 observation wells could collect at most several readings from each location a week, and samples are more often collected once a month. However, soil properties can be affected by temperature, photosynthesis, rainfall and other factors that can change on a daily or even hourly basis, and much higher spatio-temporal resolution is needed.

In this paper, we present *Suelo*, an embedded networked sensing system (ENS) designed to monitor soil with high spatio-temporal resolution. The key challenge for Suelo is data integrity: chemical soil sensors are inherently fragile and need frequent calibration and maintenance. For example, in a one week lab experiment we calibrated a set of sensors daily and found that calibration does not change linearly with time, nor does it change in a constant direction. But sensors cannot be regularly or easily removed from the ground for maintenance because doing so would disturb the soil, which can require several days or months to settle back into the original structure. At the same time, it is difficult to know from the data alone when a sensor needs repairs

SenSys'09, November 4–6, 2009, Berkeley, CA, USA.

or re-calibration; few sites have existing models of data that could be used for automatic data verification, and a sensing system generates too much data to verify each point manually. Furthermore, deploying redundant sensors and checking for consistency would not be a suitable solution because soil properties can change every few inches, and having too many sensors in one area would alter the structure, and therefore the behavior, of the soil properties being measured.

To overcome these challenges, Suelo was designed for *human-assisted sensing*: it actively requests the help of a human when necessary to validate, calibrate, repair, or replace sensors, and it learns from the human's actions and responses. Suelo is a complete end-to-end embedded networked sensing system that includes a suite of robust networking protocols (described in a more complete version of this paper [28]. In this paper we focus on the two aspects of the design that were important for human interaction. First, we converted high-maintenance sensors that were originally designed for a laboratory into an in-situ sensing platform called the *Javelin* that facilitates frequent maintenance while minimizing soil disturbance. Second, we created the Suelo Fault Detection System (SFDS). SFDS is a data integrity service that is initialized with expert knowledge about the sensors, that requests actions from the user when necessary, and that learns from the user's actions and responses. This two-pronged approach to assisted sensing allowed us to explore and discover new phenomena using unreliable sensors and hardware, without sacrificing data integrity, and while minimizing the human resources required.

Suelo was designed for *exploratory* studies, in which scientists might deploy sensors for approximately a week before moving to a new location. Every deployment is different and the nature of the data and physical relationships is not known in advance. The Suelo Fault Detection System therefore uses human assistance in two ways. First, for each new deployment it re-learns how to differentiate valid data from faulty data using human feedback, thereby reducing its dependence on carefully tuned parameters. This is particularly important for Suelo because many soil monitoring deployments are currently exploratory. Second, once the system learns to identify error states it requests repairs and maintenance from the user when necessary. In-field repairs and maintenance are necessary when using unreliable sensors, and the Suelo design facilitates this process.

All deployments must deal with faults. Given the current state of soil sensing hardware and the inescapable characteristics of soil environments, soil deployments encounter faults earlier and more frequently than many other deployments. However, we believe that many ENS deployments in the near and far future will be using early-generation hardware prototypes that will have reliability problems, and will be deployed in exploratory studies. Many of these deployments could potentially benefit from the assisted sensing principles and mechanisms used in the design of Suelo.

We evaluate Suelo in three soil monitoring deployments: the Ganges delta in Bangladesh, a forest in the James Reserve, and the junction between the Merced and San Joaquin rivers in California. The following key results are demonstrated in our evaluation:

- With human assistance, the Suelo Fault Detection System produced 91% fewer false negatives and false positives than common solutions that do not leverage human interaction.
- In our final deployment 93% of data collected by Suelo was usable, and the SFDS and hardware design allowed an estimated 10% of otherwise faulty or mis-classified data to be salvaged or correctly classified by requesting human intervention at the right time.

Our deployments required substantial human assistance, but they produced over 100 times more usable data points than scientists collected using observation wells.

2 Motivation

In this section, we explain why soil monitoring is important, why manual data collection techniques are insufficient for scientific demands, and why soil monitoring presents a particularly ominous set of challenges for remote sensing.

Soil monitoring is an important sensing application that will help answer many open scientific questions. For example, one group of scientists that we worked with has been studying the factors controlling the mobilization of arsenic into ground water near the Ganges Delta region in the Munshiganj district of Bangladesh for the past 10 years. Their recently confirmed hypothesis, published in Nature [23], states that bacteria that live in the soil and sediment are the cause. Arsenic, originally from the Himalayas, sticks to rust particles. The particles are buried several feet under ground, creating an environment with no oxygen. In such an anoxic environment, the bacteria resort to a respiration process that takes in rust and arsenic. This process transforms the arsenic into a form that dissolves in water, and ultimately is mobilized into the groundwater (Figure 1. To test this hypothesis, the scientists have been monitoring the levels of organic matter, arsenic, and a number of other chemical concentrations in the ground over a several square kilometer rice field once during the monsoon season and once during the dry season.

Another application of soil monitoring is to measure the effect when two rivers of very different water quality flow together. Agricultural runoff, such as pesticides, is suspected to infiltrate groundwater at the confluence of two distinctly different rivers: the Merced River (relatively low salinity) and the agricultural drainage-impacted San Joaquin River (relatively high salinity) [24]. A group of scientists based at UC Merced has been studying this phenomena to understand how the agricultural runoff that pollutes the San Joaquin river impacts the Merced river and riverbed just past the confluence. This study is important in designing monitoring and remediation efforts for the Merced river [13].

Soil monitoring deployments are not restricted to only instrumenting the soil, and many soil monitoring deployments measure parameters both inside and outside of the soil. In one of our deployments, scientists deploy PAR sensors, which measure photosynthetically active radiation, in order to aid in calculating the exchange of carbon between the soil and the atmosphere. SFDS has been successfully applied to detect faults in PAR and other non-soil sensors as well.

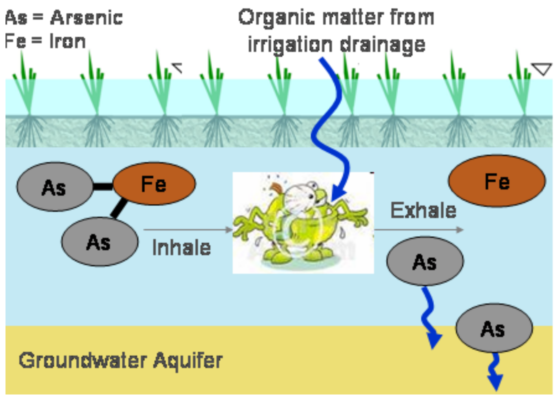


Figure 1. The Bangladesh deployment is designed to test the relationship between irrigation runoff and arsenic in drinking water. The hypothesis is that bacteria feeding on organic matter in the irrigation water break down bonds between iron and arsenic, allowing the arsenic to seep into groundwater aquifers.

2.1 State-of-the-art in Soil Monitoring

The current state-of-the-art subsurface sampling technique is called an *observation well*, which is a slotted PVC tube inserted into the ground that allows water samples to be brought to the surface using pumps. Each well is used to collect water from a single depth. These samples are then processed in an additional 10 to 20 steps, such as adding solutions and waiting or mixing the new solution for precisely-timed periods, and then they are transported to a laboratory or analyzed in the field using a mobile kit. This technique requires substantial human effort for each sample, which makes it difficult to collect data at the spatio-temporal resolution required to observe the daily or even hourly effects of temperature, photosynthesis, rainfall and other factors.

The composition of water samples collected from an observation well can be analyzed through the use of *ion-selective electrodes* (ISEs), which measure the concentration of a specific ion in a solution. Combination ISEs report the electric potential between an internal reference voltage provided by a gel probe and the ions that permeate through a selectively permeable membrane. An ISE must be calibrated by exposing it to a range of known ion concentrations and creating a function that relates the concentration levels to the output voltages of the sensor. An idealized curve for the *total detection range* (TDR) of an ISE is shown in Figure 2. The TDR can be divided into the *linear detection range* (LDR), which is the linear portion at the center of the calibration curve, and the *non-linear detection range* (NLDR). Data values in the NLDR (i.e. outside of the LDR but inside the TDR) may be valid, but are less reliable than data measured in the LDR because the NLDR has a smaller slope than the LDR, i.e. a change in ion concentration will produce a smaller change in voltage in the NLDR than the LDR.

2.2 Challenges for Unattended Soil Monitoring

ISEs are poorly matched with the demands of a long-term sensing deployment because they require frequent mainte-

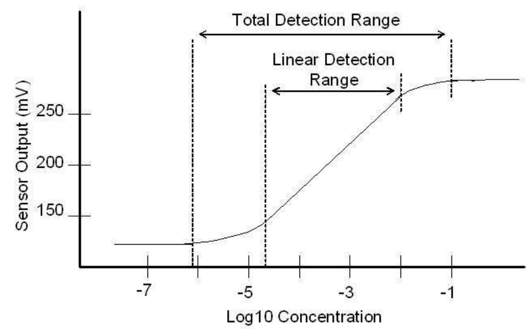


Figure 2. ISE sensors generally have a linear response to increasing concentrations of an ion (the LDR), but for extreme values they have a non-linear response (the NLDR). Values from the NLDR may need to be manually validated.

nance and calibration. The calibration coefficients of an ISE can change over time, or the membrane can become dirty, at which point it must be cleaned and reconditioned to regain sensitivity [30]. Thus, an ISE must be calibrated and checked frequently to measure sensitivity, to perform maintenance when necessary, and to eliminate faulty sensors [4]. In an embedded sensing scenario, frequent calibration disturbs the soil. Each time the soil is disturbed, it requires time for air pockets to work themselves out and the soil to compact back to the original structure. Soil settling times can take from several days to several months. But it is necessary to wait, because the presence of air and other abnormalities present in unsettled soil can significantly affect the data collected. Unfortunately, at several hundred dollars a sensor, ISEs are one of the few reasonably priced sensors. Therefore, manual calibration of ISEs is important, but should be done as infrequently as possible to keep soil disturbance to a minimum.

In some instances, denser deployments of supplemental sensors can aid in the manual calibration process. The challenge with deploying additional sensors in soil is that often the signals being measured are spatially discontinuous and can produce substantially different measurements at locations separated by only a few feet or even a few inches. Thus, the use of dense sampling to validate sensors and their calibration coefficients often require an impractically high density of redundant sensors to be effective. Not only would it be costly, but too many sensors can also disturb the signal that is being measured. For example a soil moisture sensor computes saturation levels using the change in frequency of a low powered RF signal transmitted by the sensor; multiple sensors in close proximity could interfere with neighboring readings. Another challenge is that high-resolution sensing of soil properties has only been recently possible, so there are limited pre-existing models of the stimulus that can be used to verify or calibrate the sensors. Indeed, the high-resolution data collected by the sensing deployments we describe in this paper uncovered diurnal trends that were not expected by the domain scientists and previously unknown.

Manual validation of some data is possible using physical samples, but most sensing deployments produce too much data to manually verify and check for calibration drift. The deployments we describe later in the paper collected data from 50 sensors, four times an hour. This is too much data to manually verify with traditional approaches using observation wells.

A combination of manual calibration, validation, and dense sampling can aid in improving the quantity and quality of data collected in a soil monitoring deployment, and are complementary solutions that can be used with Suelo. Our goal was to explore what else could be done to improve soil sensing deployments.

3 Related Work

Many recent ENS deployments have used unreliable sensors [4] and hardware [1, 18]. For example, a deployment in the redwoods of California, 8 out of 33 temperature sensors fail [35]. The authors of the Life Under Your Feet deployment discuss their experiences with data faults, highlighting the need for fault detection algorithms, similar to the Suelo Fault Detection System, that can be customized based on sensing modalities, and trained in the field to discern important events from actual faults [12]. These experiences are not uncommon, and will only increase, as people push the limits of sensor technology and deployment possibilities [38]. Data fidelity is critical if data is to be used for scientific purposes. In this section, we review several techniques that people have previously used to deal with unreliable sensing systems.

Several studies have used predictive models to identify when a sensor begins to produce faulty or uncalibrated data. Some use theoretical or mathematical models of the soil or physical phenomena to identify faults [9, 17]. Several studies have shown that correlations between nearby nodes can be used for in-situ sensor calibration or on-line fault detection. For example, Whitehouse et al. used distance relationships to formulate constraints on the calibration coefficients of speakers and microphones in a network of neighboring nodes [37]. Ganeriwal et al. built a *web of trust* between neighboring nodes that sense the same phenomenon to detect security attacks or faulty nodes [11]. Larkey et al. [19] and Nath et al. [21] use Naive Bayes classifiers to estimate the spatial distributions of data in order to identify faulty nodes. However, spatially discontinuous signals in soil make it difficult to apply calibration or fault detection techniques that rely on known correlations or constraints between spatially proximate sensors. Additionally, solutions such as Larkey et al., or Nath et al., assume that faults are not common, and are therefore anomalies. This has not been the case in ours' or others' experiences.

Many machine learning algorithms rely on labelled datasets as *training data* to perform clustering, classification, or outlier detection [7, 8, 10, 16]. Other approaches use historical traces captured from the system when it is known to be operating correctly to build models for different types of behavior [2, 15]. These approaches, however, can only be applied once domain scientists have collected enough data to create models or training data sets. Additionally, many of these systems assume that training data contains few or no faults, and that training data accurately represents expected behavior.

Many ENS applications – including the soil monitoring deployments described in this paper – are *exploratory* in the sense that the data has never been collected and the domain scientists do not know what to expect. SFDS' assisted feedback makes it easy for users to modify the system's assumptions and algorithms even after the deployment has begun, enabling the system to better adapt to previously uncharacterized environments.

User driven segmentation is a common practice in data mining and exploration that utilizes expert knowledge regarding the importance of certain sub-domains in a space [3]. It differs from unsupervised learning processes like clustering in that it leverages the user's knowledge to partition data into similar groups. Ramel et al. describe a system similar to SFDS that is applied to the classification of images taken from historical books [29]. The system maintains a set of rules that are updated by the user to classify each book into four possible categories. The use of rules for user driven segmentation is complementary to SFDS.

Online supervised learning systems build a model in real time, instead of relying on a labeled dataset provided in advance. Bohus et al. describe an online supervised learning system to improve spoken language interfaces [5]. When the voice recognition software either mis-understands or does not understand what the user said, the system can rephrase the question or ask the user to repeat their answer. Such systems are often best when the system can distinguish the difference between success and failure immediately and without human feedback. But in general online supervised learning techniques are also complementary to SFDS.

4 The Suelo Platform Design

Suelo is an end-to-end embedded networked sensing system that includes a suite of robust, autonomous networking protocols that provide link-layer reliability, a delay-tolerant networking layer (DTN), and an automated network monitoring system. Despite link failures and base station outages, the system still recovered 91% of the expected packets, which is high relative to other real-world ENS deployments.

We do not discuss the networking protocols due to space limitations. Instead, we focus on the two aspects of the design that were important for human interaction. First, the hardware was designed specifically to enable human intervention while minimizing soil disturbance when sensors need to be validated, calibrated, repaired, or replaced. Second, we designed an on-line data analysis tool that would actively handle sensor faults by requesting and learning from human intervention.

4.1 Hardware Platforms

To monitor chemical concentrations in soil, we converted ion-selective electrodes (ISEs) into sensing platforms that could be inserted into the ground and would transmit readings over a wireless connection in real time. We built two different hardware platforms for the two different types of sensor applications: the TapRoot and the Javelin. The TapRoot was the first-generation design that focused on electrical and wireless requirements. After our first deployment, however, we found that the unreliable nature of the ISEs also introduces requirements for human interaction during deployment, validation, calibration, and repairs. This motivated our second-generation design called the Javelin. We describe the TapRoot design in Section 4.1.1, the requirements for human interaction in Section 4.1.2, and the Javelin design in Section 4.1.3

4.1.1 1st Generation: The TapRoot

The TapRoot sensing platform was designed for scientists studying the mobilization of arsenic into ground water near the Ganges River in Bangladesh. These scientists need three types of data:

1. **ammonium**, **calcium**, **chloride**, **nitrate**, and **carbonate** concentrations
2. **pH** of the ground water
3. **oxidation-reduction potential (ORP)** of the soil

Ammonium and calcium concentrations are important because previous work has shown that they correlate with arsenic concentrations in the Ganges delta region [14]. Thus, they serve as *proxy measurements* for arsenic concentrations, for which no off-the-shelf sensors exist. The ORP sensor responds to changes in organic matter, and is therefore used to detect when organic-rich irrigation water flows through. Because ORP is not always a reliable measure, nitrate and carbonate sensors are also deployed to further characterize the organic matter. Chloride ions can interfere with nitrate measurements, so we measured chloride; pH is a measure of the acidity of the water, which controls the form that an ion will take (for example as pH increases nitrogen particles will shift from *ammonium*, NH_4^+, to *ammonia*, NH_3).

The TapRoot used seven different models of ion-selective electrodes (ISE) purchased from Sentek to produce these measurements. The output from ISEs is temperature dependent and must be calibrated for temperature using the Nernst equation [30], so the TapRoot also includes thermistors purchased from Digikey to measure the temperature of the soil. ISEs are designed for measuring water chemistry and not necessarily for soil environments, but we experimentally demonstrated in the laboratory that ISEs can be used for soil monitoring when saturation levels are high enough. Therefore, we also deployed moisture sensors purchased from Decagon in each TapRoot and 2 pressure transducers (not integrated with the TapRoot because they had local logging capabilities) to interpret the output of the ISEs.

At each sensing location, the entire sensor suite of 8 sensors was deployed at 3 different depths to characterize the chemistry above, below, and in middle of an iron band that the domain scientists believed to source the arsenic mobilization. Thus a total of up to 24 sensors was deployed with each TapRoot platform.

The TapRoot uses a PVC enclosure that can house 4 mica2 motes and the corresponding ADC boards needed to convert the analog sensor inputs to digital readings. Each ADC board could support up to six sensors, so up to four mica2 nodes were required in a TapRoot when deploying the maximum of 24 sensors at a site. The mica2 motes periodically sample the sensors and transmit the data to a basestation. The mica2 enclosure sits on top of a column as shown in Figure 3(a) in order to elevate the radio to avoid RF attenuation due to ground water.

Cables run from the enclosure into the ground to the sensors, which are placed directly in the ground. To deploy the sensors, we dug three holes per depth to accommodate a full suite of sensors: 4 ISEs in one hole, 3 ISEs in another, and the moisture sensor in a separate hole so that their electromagnetic radiation would not interfere with the electric potential measured by the ISEs. Thus, to deploy at three depths, 10 holes needed to be dug, including one for the column to hold the PVC enclosure. Sensors were deployed directly into the mud for maximum contact (Figure 3(a)).

To sample from an ISE sensor, the mote collects 15 readings at one second intervals, drops the first five readings, and averages the last 10 readings in order to minimize the noise from the sensor. This approach was derived from experimentation in the lab to reduce noise in the ISE readings.

4.1.2 Human-oriented Hardware Requirements

Our experience with the TapRoot generated three design principles for a sensing platform using unreliable ISE sensors. The hardware platform should:

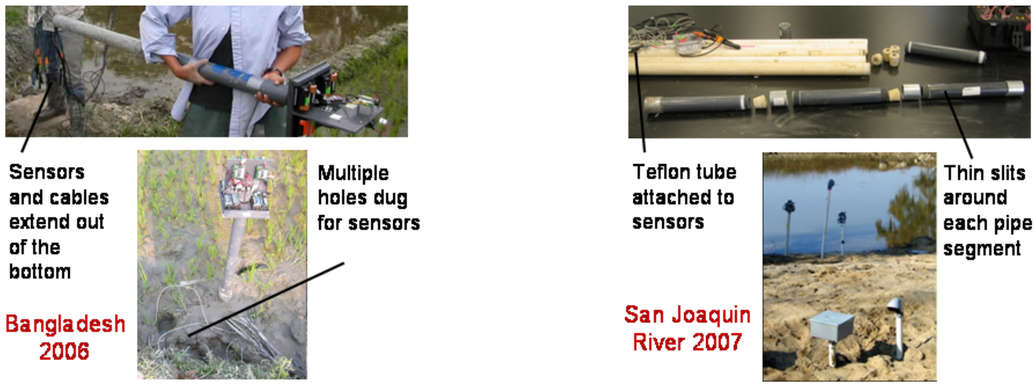
1. facilitate the extraction of physical samples near sensors for data validation;
2. support in-situ calibration, testing and replacement of individual sensors; and
3. be quick to deploy or re-deploy, and should minimize the impact on the soil to keep soil settling times at a minimum.

During our deployment more than 10% of our data (primarily from the nitrate and chloride sensors) needed to be manually verified by collecting and analyzing a supplemental water sample. This would have been easier and increased our confidence in the data if the TapRoot were designed to facilitate sample extraction from the immediate location where the sensor was deployed.

We also observed a number of faulty sensors in Bangladesh. For example, within the first several days, we were sure that at least one calcium sensor, and possibly all of the oxidation reduction potential sensors, were not working. We wanted to be able to remove the sensors to test them in the mobile chemistry lab we had set up in a tent. However, the TapRoot package was not designed to facilitate maintenance or redeployment. Replacing a sensor required pulling all 24 sensors out of the ground. (Re)deploying an entire TapRoot took all day because we had to dig individual holes for each pair of sensors. Moreover, the weight and unmanageable length of extra cable that is required make a fully loaded TapRoot extremely difficult to maneuver, especially when tromping through knee deep mud. Deploying and redeploying the TapRoot was too difficult to repair or replace even a single sensor.

4.1.3 2nd Generation: The Javelin

The Javelin platform was designed to address the problems of the TapRoot. The basic design is a single 1.25 inch PVC tube that houses all sensors and communication hardware, as shown in Figure 3(b). The Javelin narrows at the bottom so that it can be driven into the ground, minimizing the impact on the soil and avoiding the need to dig holes for the sensors or for the pylon structure itself. A pole pounder



(a) The TapRoot platform, as used in Bangladesh, 2006 (b) The Javelin platform, as used in San Joaquin, 2007

Figure 3. The first-generation TapRoot platform is unwieldy to carry and difficult to deploy, requiring multiple holes to be dug. The second-generation Javelin platform makes it easier to validate, calibrate, repair or replace the sensors by putting all sensors into a single column that is easily driven into and removed from the ground, minimizing soil disturbance.

is used to drive the outer PVC pipe into the ground. Once the pipe is at the correct depth, the Javelin is slid into the pipe, which serves as a sleeve, and the pipe is pulled out. Deploying a Javelin takes under an hour in harder soils; in the soft mud of Bangladesh it would take much less. A Javelin is small and is not designed to handle more than 5 sensors, making it easy to maneuver.

The Javelin has slits around the circumference of the tube to allow moisture in, but keep out soils and other particles that may damage the sensor membranes. It also contains Teflon tubes that extend from the top of the pylon down to each sensor. These Teflon tubes act as observation wells, and can be used to extract physical samples from near the tip of the sensor in order to validate questionable data. Thus, the Javelin combines automatic yet unreliable sensing using ISEs with labor-intensive yet reliable manual sensing using observation wells. The result is an automatic yet verifiable data stream. These Teflon tubes can also be used for in-situ calibration by dropping known solutions into the tubes, essentially running the observation well in reverse.

The main limitation of the Javelin is that it would not perform well in environments that are not moisture saturated, since the sensors are shielded by the column and would not come into contact with sufficient moisture. In saturated soils such as those we studied in Bangladesh or the San Joaquin River, or even semi-saturated soils, the Javelin performs well.

We built the Javelin for scientists studying the impacts of agricultural run-off near the Merced and San Joaquin rivers in California. Therefore, each Javelin contained up to 2 ammonium sensors, 2 nitrate sensors, and 2 temperature sensors. Nitrate is one of the chief ions present in the agricultural pollutants, and ammonium is a different form of nitrate. We deployed sensors for both of these ions to identify the presence of recent or past agricultural pollutants. The Javelin has since been used in Bangladesh and with numerous other deployments.

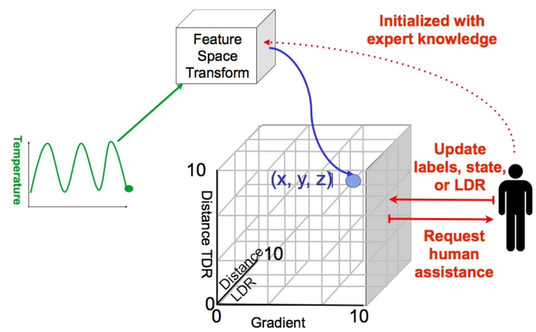


Figure 4. SFDS uses a feature space transform defined by an expert user to identify faulty data, and suggests actions to the user based on the region in feature space where a point falls. User feedback helps the system improve its fault detection parameters and action recommendations.

4.2 Human-assisted Data Assurance

A key component of Suelo is the Suelo Fault Detection System. The SFDS automatically analyzes data and notifies the human assistant when a sensor needs to be validated, calibrated, repaired, or replaced. The overall architecture of this system is illustrated in Figure 4. The system is first initialized by an expert, who defines features of the data that are likely to correlate with sensor failures. Sensors transmit data at periodic intervals to a base station over their wireless radio. Upon the arrival of new sensor data, SFDS extracts the features from each data point and maps the resulting feature vector to a feature space. An outlier detection algorithm differentiates between valid and faulty system states in the feature space. The location of a feature vector in the feature space is also used to identify a recommended action for the human assistant. Suelo notifies the assistant of the action to take in real-time. Once the assistant takes an action, they can provide feedback about data validity or sensor repairs, which

SFDS incorporates into the data assurance system to improve future recommendations. These mechanisms are discussed in more detail in the following sections.

4.2.1 Initializing with Expert Knowledge

The data assurance system must be initialized by an expert with domain knowledge pertinent to the specific sensors and environment being analyzed. Specifically, the expert must define a *feature space transform* that will help identify invalid data points. A feature space transform is a set of functions that can be called on a data point or set of data points to help elucidate certain properties about the data. Domain experts in soil monitoring and ISEs helped to define the following feature space transform from (v_i, t_i) to (x_i, y_i, z_i) for data coming from soil sensors:

$$x_i = |(v_i - v_{i-1})/(t_i - t_{i-1})| \quad (1)$$

$$y_i = \max(v_i - LDR_{upper}, LDR_{lower} - v_i, 0) \quad (2)$$

$$z_i = \max(v_i - TDR_{upper}, TDR_{lower} - v_i, 0) \quad (3)$$

where (v_i, t_i) are the value and time-stamp for the i'th data point, and $(LDR_{upper}, LDR_{lower})$ and $(TDR_{upper}, TDR_{lower})$ are the upper and lower bounds of the linear detection range and total detection range, respectively, for a particular sensor, as determined through the calibration process or by looking at the sensor's specifications sheet.

Each feature or combination of these features captures a potentially different type of error that might occur. The GRADIENT feature, defined as x, is the discrete derivative, which is calculated as the change in value over one time step. This feature captures the intuition that diffusion phenomena obey physical limits and readings should not change very quickly. The DISTANCE LDR feature, defined as y, is the distance between a data point and the sensor's LDR. This feature captures the intuition that data points become less reliable as they are farther from the linear detection range of the sensor, and may need to be manually verified. The DISTANCE TDR feature, defined as z, is the distance between a data point and the sensor's TDR. This feature captures the intuition that data points outside the detection range of the sensor likely indicate a problem with the calibration or the sensor itself.

SFDS requires the domain scientists to create the feature space transform such that faulty data have large feature values, and therefore fall far from the origin of the feature space, and non-faulty data have small feature values, and therefore fall close to the origin of the feature space. In other words, as the feature value increases, the probability that the sensor producing that value is faulty also increases. The expert must also provide a scaling function for each feature so that, by convention, a value of 0 represents non-faulty data, and a value of 10 or more represents almost certainly faulty data. Given a transform with these properties, SFDS can detect faults using a simple outlier detection algorithm that identifies points that lie too far from the origin. The scaling functions need not be exact, and the outlier detection algorithm is robust to a large range of scaled values. The primary function of the scaling function is to help with system initialization. With the help of domain scientists, we defined a scaling function $f(n) = \max(\log_2 n/S, 0)$. $f(n)$ is applied to the three feature values (x, y, z). S is chosen for each feature such that $\log_2 n < S$ for most non-faulty data points. The scaling function for each feature is listed in the right column of Figure 5.

Dimension	Feature	Scaling
x	GRADIENT	$\max(0, \log_2(n/16))$
y	DISTANCE LDR	$\max(0, \log_2(n))$
z	DISTANCE TDR	$\log_2(n)$

Figure 5. An expert user defines a scaling function for each dimension of the feature space so that SFDS can differentiate between good and faulty data during the initialization period.

Finally, the user must initialize the system with an action or set of actions that should be taken when a faulty point is detected in each region of the feature space. SFDS divides the feature space into 27 equal sized regions. Each region is *labelled* with an action that an assistant should take if a data point is mapped to that region in the feature space. The same label is assigned to all points in a region. The rationale is that faulty points that lie close together are likely remedied by the same action. We used the expertise of domain scientists to assign one of four possible labels to each region:

1. validate questionable sensor data
2. check the sensor/environment
3. re-calibrate a sensor
4. replace a sensor

For example, a point that maps to the feature vector (9, 1, 1) has a high value for the GRADIENT feature, and low values for the other two features. This means that the sensor reported a sudden and dramatic change in value, but is still reporting data within the expected range. The domain expert's intuition tells us that the environment should be checked for a recent event (e.g. perhaps the field is being irrigated), or the sensor should be checked to find a fault (e.g. maybe the sensor cable was disconnected). Therefore, the expert assigns the region containing this point the action: "check the sensor/environment". Because certain actions, such as recalibration or validating questionable data, are more labor intensive, the expert is careful in assigning labels. These labels can also be updated by the assistant in real-time, as described below.

4.2.2 Requesting Human Assistance

The data assurance system applies an outlier detection algorithm to the feature space to identify faulty data points. It then uses the data point's location in feature space to recommend some action to the human assistant.

To identify faulty data points, when the system receives a data point from a sensor it calculates the feature vector as described above. It then calculates the magnitude of the new feature vector:

$$d_i = \sqrt{x_i^2 + y_i^2 + z_i^2} \quad (4)$$

Initially, for some initialization period P, all points i for which $d_i > 5$ are identified as faulty and all other points are non-faulty. Over time, the system estimates a distribution of non-faulty points according to the following equations:

$$\mu_i = (1-\alpha)d_i + \alpha \cdot \mu_{i-1} \quad (5)$$

$$\sigma_i^2 = (1-\alpha)(d_i - \mu_i)^2 + \alpha \cdot \sigma_{i-1}^2 \quad (6)$$

$$\sigma_i = \sqrt{\sigma_i^2} \quad (7)$$

In other words, SFDS uses an EWMA-based Gaussian estimator to estimate the parameters (μ_i, σ_i) of non-faulty data points as a Normal distribution at time i. After the initialization period P is over, SFDS identifies all values with $d_i > \mu_i + 2\sigma_i$ as faulty. The default value for P is set to 300; the default value for α is set to 0.9.

Once a data point has been identified as an outlier, SFDS uses the point's location in the feature space to suggest an action the user should take to address the problem. Each feature vector maps to some region in the space. SFDS looks up the region to which the feature vector maps, and notifies the user of the label associated with that region. The human assistant might either be available at the site or working remotely. Most exploratory deployments have at least one human who is always at hand. The assistant can take an action, if they are present at the deployment site.

This feature space, outlier detection algorithm, and labeling system were designed to make it easy for an expert to initialize, define actions, and provide feedback over time. Note that this is only one of many outlier detection algorithms possible, and many different outlier detection algorithms are currently being explored. For example, stream clustering-based outlier detection algorithms can be initialized and can be given feedback from the user, but the user cannot necessarily associate actions with a cluster because the cluster sizes, location, and contents may change over time. The contribution of this section is not the outlier detection algorithm itself, but rather the use of outlier detection to manage human resources during a deployment with unreliable sensors.

4.2.3 *Learning from Human Responses*

An onsite or remote human assistant can provide information to SFDS at any time in two ways. First, the user can provide information about the validity of data. Although the feature space makes fault detection simple, instances arise when data that fall far from the origin are not faulty, or vice versa. For example, an unusually heavy rain may lead to unexpectedly low concentrations of certain ions in the soil. In order to tell the system about such occurrences, the user requests a snapshot of the feature space using the command line interface. The output is printed to the screen. A small sample of this output is shown in Figure 6. Each region in the space is represented by the feature vector located at the region's center, the label associated with that region, and additional metadata. Below each region, the snapshot includes a list of the most recent data points associated with that region. Using the command line interface, users can update a label or fault state associated with a region. This information can be applied to all future data that is associated with that region, or with data from a specific sensor. For example, on one particularly windy day, wildly waving tree branches cast quickly moving shadows on the PAR sensors, which measure light. As a result, most of the PAR data was extremely

Cluster ID	x (gradient)	y (distance LDR)	z (distance TDR)	Faulty	Action	
0	3.3	6	6	YES	RECALIBRATE	
Node ID	x	y	z	Data	Timestamp	Sensor
18	2.5	5.4	4.5	54	1171915720	TEMPERATURE
16	2.3	6.2	5.3	135	1171915715	AMMONIUM
12	3	6	5.8	254	1171915715	NITRATE

Cluster ID	x (gradient)	y (distance LDR)	z (distance TDR)	Faulty	Action	
1	0	0	0	NO	NONE	
Node ID	x	y	z	Data	Timestamp	Sensor
15	0.1	0.5	1	100	1171915720	CALCIUM
16	0	0	0	54	1171914000	TEMPERATURE
12	0	0	0	10	1171914390	MOISTURE

Figure 6. Suelo responds to queries with an output of the status of the feature space as shown above. Each region in the feature space is represented by a feature vector, the fault status for that region, and the label (outlined in red in the figure). The set of points located in each region, printed below the region header, is represented by a feature vector, the sensor data, timestamp, and sensor type (outlined in black in the figure). The figure is a snapshot of the status for two regions.

noisy that day, and was classified in two regions in the feature space with high *rate_of_change* values. Suelo notified the researchers to check the PAR sensors. After returning from the field, using the command line interface the researchers updated the feature space to classify PAR data as not faulty in the two regions with high *rate_of_change* values. We evaluate the impact of user feedback on detection accuracy in our evaluation.

5 Evaluation

We performed three deployments with Suelo in Bangladesh, the James Reserve, and the San Joaquin river. We evaluate this system in two ways. First, we show that the number of *misclassified data points* are lower with the Suelo Fault Detection System than with two baseline systems that do not leverage human interaction: threshold-based fault detection, and rule-based fault detection. Second, we demonstrate that much the reason for this improvement is due to human actions requested by the system.

To compare these systems, we need to know what data is truly faulty, what data is truly not faulty, and when any faults occurred. In other words, we need access to ground truth. In a simulation, ground truth is precise because we can inject a known fault at a known time into the simulation. Attaining ground truth in the field, especially for environmental data, is not as straightforward. Our exploratory sensing deployments collected data about environments where little is known about the exact chemistry and daily biological reactions. In these instances, an approximation to ground truth is achieved using a combination of manual analysis by domain experts, results of physical samples that were extracted during the deployment and analyzed in a lab, and post-deployment analysis and calibration of sensors. We treat the results of this analysis as ground truth when evaluating accuracy for environmental sensor faults.

5.1 Deployments and Data Collection

This evaluation uses data sets collected from three deployments. The first data, which we refer to as *Bang*, was

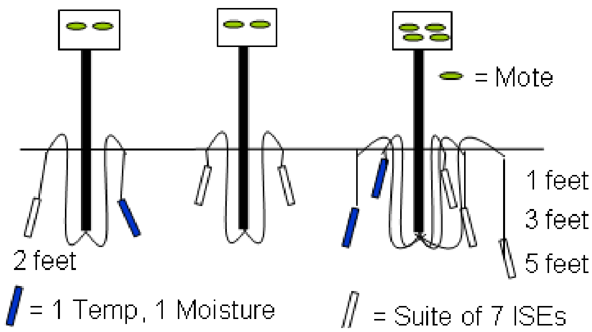


Figure 7. The TapRoot was used in Bangladesh to measure soil at multiple depths. Each sensor was buried individually, requiring many cables and making it difficult to perform maintenance during deployment without disturbing the soil.

collected from a deployment undertaken in a rice paddy in Bangladesh in January, 2006 to help scientists evaluate the relationship between irrigation and arsenic contamination in the groundwater [25]. The experiment was designed and deployed with scientists and civil engineers from the Bangladesh University of Engineering and Technology and MIT. Our field site in Bangladesh consisted of a series of rice paddies separated by irrigation troughs built of mud and clay from the field. This deployment has been described throughout the paper, so is not described again here. The layout is shown in Figure 7. The network collected 26,000 measurements over a period of 12 days from 42 ISE and temperature sensors.

The second dataset, which we refer to as *JR*, was collected from a deployment at the James Reserve in September, 2007. The purpose of this deployment was to explore the spatial and temporal scales at which sub-surface measurements should be taken, and to study the relationship between soil CO_2 fluxes, moisture and temperature conditions in the soil, and to study the relationship between micro-scale and macro-scale ecological processes. Above ground, air temperature, relative humidity, barometric pressure, and photo-synthetic active radiation (PAR) are measured. Below ground, temperature, moisture, and CO_2 concentration measurements are taken at depths of 2 cm, 8 cm, and 16 cm in the soil. These sensors, connected to Hobo dataloggers with a wired upload channel, have been in the ground collecting data since October, 2005. Before our deployment, scientists would collect data by connecting a laptop to the dataloggers once every month or two. We connected the Hobo dataloggers to wireless Mica2 motes so that scientists could collect the data in real time, and more importantly, identify and fix faults. The sensing system spans approximately 80 m and consists of 10 sites, with 13 sensors at each site. We collected 35,000 measurements during our 1 day field visit.

The third dataset, which we refer to as *SJR*, was collected from a deployment at the junction of the San Joaquin River and the Merced River in February, 2007. This deployment was described in Section 2. We deployed 14 ammonium and nitrate ion-selective electrodes and 7 temperature sensors connected to Mica2 motes. 12,000 measurements were collected over this 5 day deployment. We systematically validated data collected from all 21 sensors to ensure that SFDS did not miss any faults, and did not direct us to take any unnecessary actions in the field. We took two steps to validate data collected from sensors. First, we deployed a second set of sensors to shadow the scientist's main deployment. However, as already described, the heterogeneous nature of soil makes it virtually impossible to provide true measurement redundancy in these environments. We periodically extracted water samples from the Teflon tube at each Javelin to obtain an independent measurement of the ammonium and nitrate concentrations. Water samples were analyzed using a Hach Kit, which is a mobile spectrometer designed to analyze samples in the field. Because pollutant concentrations can vary during the day, we extracted samples at the beginning and end of each of the five days from all seven sampling sites. Extracting and analyzing even a single sample requires two people working in parallel for half an hour, so our six person team spent most of the deployment either validating sensor data, deploying sensors, or testing sensors.

We execute SFDS on the data from each of these deployments.

5.2 Baseline Solutions for Comparison

We compare SFDS to two baseline solutions that might otherwise have been used to detect faults in a stream data from of ISE sensors: a generalized thresholding scheme and a more complex set of classification rules. Thresholding is perhaps the most common technique used to identify the presence of faulty sensors, and has been used by several ENS deployments in the past [32,33,35]. We implement two common thresholding techniques. The first approach applies static ranges to data values: $Range_{upper} = \theta \cdot range_{upper}$, and $Range_{lower} = \theta \cdot range_{lower}$. The second approach dynamically calculates a range using running estimates of the mean (μ) and standard-deviation (σ): $Range_{upper} = \mu + \theta \cdot \sigma$, and $Range_{lower} = \mu - \theta \cdot \sigma$. For each algorithm, we identify the θ that produces the highest accuracy, and only show the results for the best performing algorithm. The first algorithm performed best for JR (with a parameter setting of $\theta = 2$), The second algorithm performed best for Bang ($\theta = 0$) and SJR ($\theta = 3$).

Of course, thresholding systems cannot detect all types of faults, and after several deployments a domain expert may be able to generate a more sophisticated system to identify faulty data. Indeed, Bertrand-Krajewski et al. [4] develop a set of seven criteria derived from physical processes underlying the data and measurement system to determine validity of data collected from ISEs. These criteria are: automatic status report from the sensor, detection range for the sensor, expected range given local phenomena, duration since last maintenance, discrete derivative, physical sensor redundancy, and expectations calculated from a model. If a data point fails a single criterion, the point is considered unreliable.

As Bertrand-Krajewski et al.state in their paper, not all of the rules apply to all of our datasets. Because the deployments were exploratory, we did not have a model or a method to calculate an expected range; our sensors did not provide status reports; we do not have a model for sensor degradation

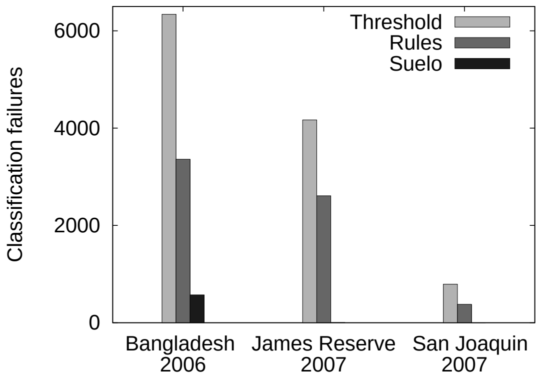


Figure 8. When tested on empirical data from three different soil monitoring deployments, SFDS produces 91% fewer false negatives and false positives than common solutions.

	Bang	JR	SJR
#Sensors	33	21	130
#Data	14854	35414	9775
#Faults	8000	3832	705
True Negatives	98%	100%	100%
True Positives	94%	100%	100%
Number of updated labels	2	4	0

Figure 9. SFDS was used on several deployments with very different sensors, sizes, durations, and fault rates, and SFDS performed well in each of these situations.

over time. Independently of that work, we identified a complete list of the fault types that were common in our deployments [22]. Then, we adapted a subset of the rules described by Bertrand-Krajewski et al. to detect each of the following fault types:

1. Invalid Data Range: when a value exceeds some predefined maximum or minimum value
2. Rapid Change: when a value changes too quickly
3. Noise: detect a fault when a standard deviation grows by some factor
4. Uncertain Data Range: when a value is in the non-linear detection range

This set of rules was actually an early version of the SFDS design, and was thus designed with rigor and completeness. We stopped developing and refining these rules when they ultimately decided that a rule-based system would never suffice for on-line assistance of unreliable sensors like ISE, for reasons described and demonstrated later in this evaluation.

During the evaluation, we optimized the parameters for the rules and threshold system using the same data that it would be tested on, i.e. we choose the set of parameters that provide the best total classification rates over all three deployments.

5.3 Results: Data Classification Accuracy

We define *classification failures* to be the number of data points that are classified incorrectly. This includes good data that are incorrectly labelled as faulty (false positives) and faulty data that are incorrectly labelled good (false negatives). Classification failure is the sum false positives and false negatives. The goal of SFDS is to reduce classification failures.

We compared SFDS to baseline solutions by running each of the three algorithms on the data sets produced by each of our three deployments.

The results are summarized for all three algorithms in Figure 8, and shown in detail for SFDS in Figure 9. Our results show that SFDS produces fewer misclassified data points than either basic thresholding techniques or a specially designed set of rules. The threshold system misclassified 11,300 points. The rule based system generally performed better than the threshold system, and only misclassified 6350 points. This is because the rule based system could identify faults where the noise levels or the values changed quickly, and uses thresholds informed by expert knowledge. However, SFDS consistently outperformed both baseline solutions, mis-classifying only 580 points in total, which is 91% fewer false negatives and false positives than the rule-based solution. However, SFDS does not detect any *class* of errors that is not explicitly targeted by the rule-based fault detection system. They both detect two basic types of errors: those where the value is out of a particular range, and those where the value or noise changes very quickly. Thus, the question arises: why does SFDS outperform the rule-based system.

5.4 Analysis: The Effect of Human Assistance

There are two ways in which human assistance improves the system accuracy: a human either takes an action in the field or provides feedback to Suelo. Sometimes both types of actions are needed together; in one instance a scientist extracted a sample to verify questionable nitrate readings, and then updated Suelo that the nitrate data was actually good. We evaluate the impact of both types of human assistance in the next two subsections, respectively. In the remaining subsections, we provide examples of each type of action that the user performed in the real deployment.

5.4.1 The Effect of Human Feedback

Our analysis shows that human feedback after manual data validation is a key reason why SFDS outperforms the baseline fault detection solutions. Figure 10 shows the results of SFDS as the amount of user feedback increases. Without any user feedback, SFDS performs very similar to the rule-based system; the rules even perform slightly better than Suelo for data from James Reserve (center bar in the figure). However, once user feedback is incorporated, the accuracy of SFDS increases as well.

SFDS leverages the user feedback to tune parameters during the deployment, reducing its sensitivity to initial parameters and even allowing it to use different parameters for each node when the environment demands. In all of our experiments, SFDS uses the same set of initial parameters. In a more complete version of this paper, we show through an

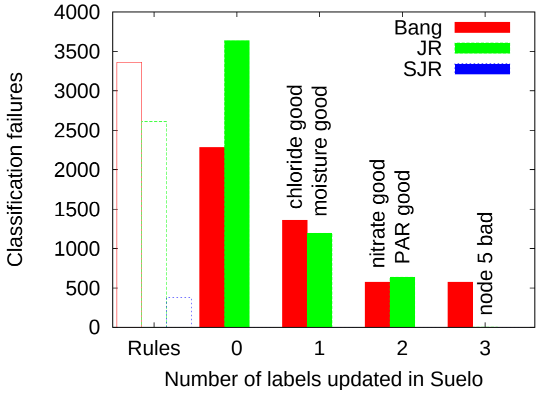


Figure 10. SFDS does not initially perform better than the rule-based solution. However, once the user updates the labels in the data assurance system, as shown in the text in the graph, the performance improves. For example, once the user verifies the chloride data by taking a sample, they notify the data assurance system that chloride data in a certain region is not faulty. This updated label reduces the number of misclassified points by almost 1000.

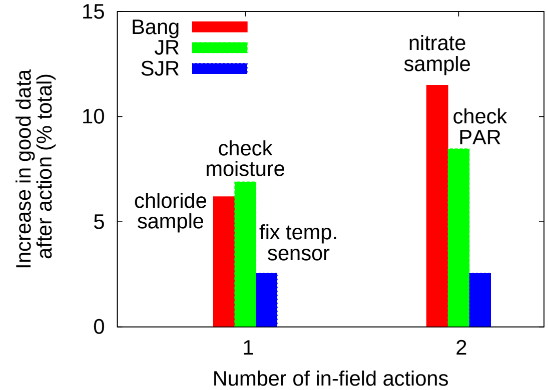


Figure 11. The X-axis is the number of times Suelo requested human assistance in the field and action was taken. The Y-axis is the increase in good data, represented as a percentage of the total amount of data, after the action was taken. The user actions to calibrate, repair, or replace sensors improved the amount of good, usable data for all three deployments.

extensive sensitivity analysis that SFDS' results are robust to scaling factors, the initialization distance and training duration d_I and P, and the number of regions used to divide the feature space [28]. Further, SFDS does not impose assumptions on the frequency of faults. However, there is a dependence on where the faulty feature vectors lie in the feature space.

In contrast, the baseline solutions are highly sensitive to the initial parameters and we needed to optimize the parameters of the rules and thresholding systems for each deployment in order to obtain comparable results.

5.4.2 The Effect of Human Actions in the Field

In our deployments, scientists took actions such as recalibrating sensors, validating questionable data, and fixing broken sensors. Suelo's goal is to direct users to take only the actions necessary to maximize the amount of good data received. Figure 11 shows the percent that the amount of *good* data increases after the first two actions scientists took in each deployment. Just these first two actions in the field increased the amount of *good* data across all three deployments by 8.2%.

When Suelo recommends an action, it does so for almost all faults immediately: for SJR and Bang it detects all faults immediately; for JR it detects all but 3 faults immediately (the remaining 3 faults are detected within 600 seconds, the data transmission periodicity). One reason for Suelo's rapid response in these deployments it that the features rely on at most one point in the past, and so of those faults that are detected, the detection occurs quickly. This latency analysis is not complete because low network yields will impact the true detection latency, and these issues will not show up in this analysis because such data never arrives at the base-station.

In the following sections, we provide anecdotes that depict the types of actions a human might be asked to perform, and explain the consequences. SFDS can differentiate between validation, calibration, and repair actions based on a fault's location in the feature space and feedback from the user about previous faults that had the same feature space signature.

5.4.3 Manual Data (In)validation

One of the key actions that Suelo requests is that a human validates or invalidates data. Feedback from these processes is used to dynamically adjust the parameters by which the system differentiates between valid and suspicious data. To validate sensor readings, water samples are drawn, delivered, and tested in the lab.

The top graph in Figure 12 is a graph of nitrate data collected from 3 sensors in Bangladesh. Almost all of the data is outside of the linear detection range, an indication that the data is likely faulty. In order to determine if the data were usable, the scientists we were working with in Bangladesh extracted several physical soil samples for lab analysis. We used the results from this analysis in conjunction with a computer model of the soil chemistry for that region to confirm that the levels for nitrate and chloride were good, even though they were in the NLDR of the sensor. The nitrate concentration was simply lower than the sensitivity of the sensor, so the readings appeared within the NLDR of the sensor. As a result of this lab analysis, the scientists were able to use this data. Of the 3400 data points recorded in the NLDR of a sensor, 1700 of these points are from either nitrate or chloride sensors; i.e. half of the data in the NLDR are from sensors measuring concentrations that we expect to fall in this range.

In other cases, however, data in the NLDR is in fact faulty. The bottom graph in Figure 12 shows chloride data collected from 1 sensor in Bangladesh that exhibits diurnal variations similar to those of other non-faulty sensors. The main difference is that the data produced is in the NLDR instead of the LDR. Suelo indicated that the sensor needed validation and after analyzing soil samples in the lab, scientists determined

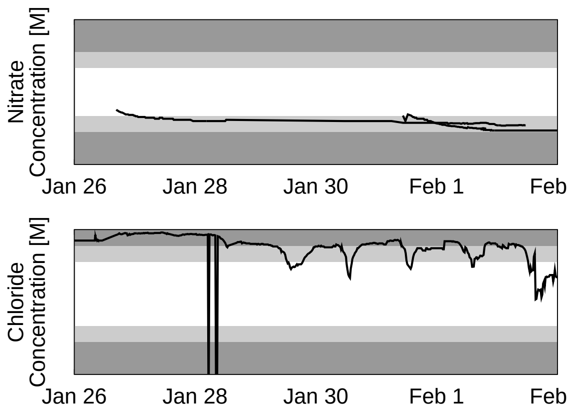


Figure 12. Data that appears faulty and is good or vice versa. LDR region is white, NLDR is shaded light gray, and outside the TDR is shaded dark gray. Top: Nitrate data taken from three different locations; potentially faulty but lab analysis reveals sensor is good. Bottom: Chloride data taken from a single location; Regular diurnal variations lead scientists to believe data is potentially good, but lab analysis reveals data is likely faulty. In all graphs containing sensor data, the white regions indicate the linear detection range (high-precision range), the lightly shaded region indicates the nonlinear detection range (lower-precision range), and the dark region indicates the remaining space.

that this data was in fact faulty.

Across all three deployments, Suelo requested 14 validation actions. 5 of them validated data from sensors that would have otherwise been classified as faulty; 9 of them confirmed the initial classification that the data was faulty. This feedback from the user resulted in 1700 readings becoming classified correctly that otherwise would have been misclassified.

5.4.4 Manual Calibration

The calibration coefficients of ISEs can drift over time as the membrane becomes dirty, and capturing this drift is necessary to interpret data. However, there are conflicting interests that make it difficult to decide when to recalibrate a sensor. In a one-week lab experiment we calibrated a set of sensors daily and found that calibration does not change linearly with time, nor does it change in a constant direction. This experiment argues for calibrating sensors as *frequently* as possible in order to capture calibration parameters. However, calibration itself is labor-intensive and causes soil disturbances that can can last from days to months. This argues for performing calibration as *infrequently* as possible.

In a particularly bad example, sensors connected to mote 11 in our Bangladesh deployment averaged a change in the calibration offset of 100 mV when comparing calibration equations obtained before and after the deployment. Given the average operational range for a sensor of 300 mV, an offset change of 100 mV is a significant change. Scientists had to discard all of the data from these sensors because the calibration had changed so significantly.

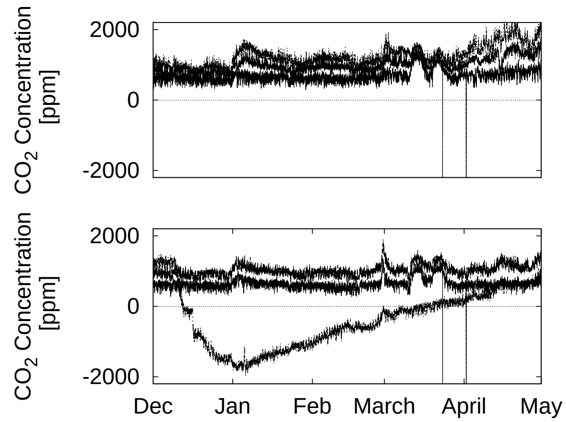


Figure 13. The calibration coefficients of an ISE sensor can drift, causing erroneous data. One of these CO_2 sensor in James Reserve gradually dips below the minimum concentration of 0.

The top and bottom panels in Figure 13 are each plots of three CO_2 sensors buried at 2, 8, and 16 cm. The CO_2 sensor's operational range is defined as concentrations above 0 ppm (indicated by the horizontal line on the graph). The top panel is representative of readings taken from 9 of the 10 locations where CO_2 sensors were deployed, which remain above this threshold. The bottom panel contains data collected from one node in the deployment where the sensors at 2 and 8 cm were good, but the readings from the sensor at 16 cm slowly dip below this line between December 2005 and March of the following year. This steady trend is not characteristic of a faulty sensor, and likely indicates that the calibration for the sensor was gradually drifting. Without further measurements taken during the time of drift, it is nearly impossible to identify the change in calibration parameters.

Across all three deployments, Suelo requested 42 calibration actions. Based on post-calibration results of the sensors after the deployments, 32 of the sensors identified by Suelo should have been calibrated earlier. However we could not remove these sensors and calibrate them because they were part of the TapRoot platform, which could not be modified in the middle of the deployment.

5.4.5 Manual Sensor Repairs

Many times, a sensor is physically broken and needs to be manually repaired or replaced. This is particularly common with exploratory deployments with first or second generation hardware platforms, and in harsh environments such as soils.

The top and bottom graphs in Figure 14 are taken from the same ammonium sensor at two different times. The top graph is of data collected in Bangladesh where a fault is apparent towards the end of the deployment. In order to reproduce the fault we deployed the sensors after returning (bottom graph) and monitored the data using an early version of SFDS. We discovered the cause of the problem to be a short in the wiring. Data readings revert to within range temporarily after we adjusted the wire at those times indicated by ◯ on the graph.

Suelo also notified us of a fault on a temperature sen-

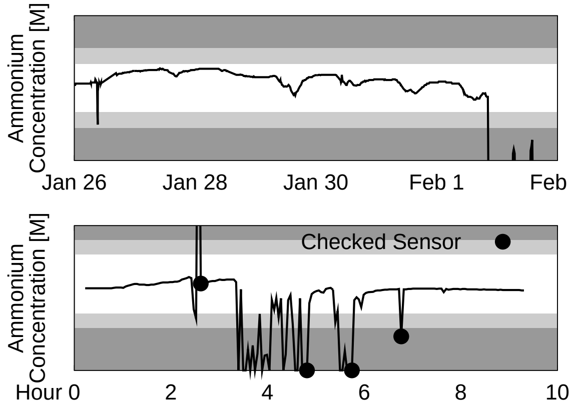


Figure 14. Actions in the field reveal source of fault. Top and bottom graphs contain data from the same ammonium sensor. Data in top graph was collected in Bangladesh, and data in bottom graph was collected after returning. Circles in the bottom graph indicate points in time when we checked the sensor and fixed the wiring. Top graph taken from a previous paper [26].

sor located next to the site where we were working. The sensor was reporting temperature 10 degrees higher than the surrounding sensors. After checking on the sensor, we immediately discovered that it had become unearthed and was exposed to direct sunlight. This fault was likely an unintended consequence of our work at the neighboring site, and the real-time response of Suelo helped us fix and repair this problem before leaving the site, and before a significant amount of data was lost.

Across all three deployments, Suelo requested 191 check sensor actions. In 75 instances, we later found that the sensor required calibration or further validation.

Many of the true positives could possibly have been fixed if the authors had the hardware resources available to replace or repair the sensors that were identified as being faulty. Thus, SFDS could potentially have a bigger effect than is shown in Figure 8.

6 Scientific Value

The most surprising discovery from this deployment was the diurnal variations observed in ammonium (representative data from one ammonium sensor shown in top graph in Figure 14). While data flattened around day 7 as a result of a scheduled irrigation event (and a sensor fault in this specific ammonium sensor), the diurnal trends (also seen in hydraulic parameters) indicate that diurnal, possibly plant-induced, processes may be important in the mobilization of arsenic. The scientists have returned to this field site twice each year to further study this phenomenon. They have recently validated the hypothesis that arsenic is mobilized near the surface [23], but still do not know what, if any, role diurnal phenomena play in this process.

Interestingly, the most surprising discovery at San Joaquin was also diurnal trends, though this time in nitrate data. The scientists are unsure about what could be causing these trends, especially because a second array of sensors just a few meters away showed no such fluctuations. Others have noticed similar patterns in river nitrate and suggested that this may have been caused by photosynthetic activity [31]. However, the diurnal behavior here is in the sediments beneath the river and the peaks are synchronized, suggesting that a sudden fluctuation in river water concentrations is not the cause.

7 Future Work: In-Situ Calibration

We are experimenting with *in-situ calibration* in order to capture changing calibration parameters while the sensor is buried in the soil. A Teflon tube is attached to a sensor, with one opening of the tube positioned just above the sensor membrane and the other end exposed above ground. Periodically, the sensor is spiked through this tube with several milliliters of a standard solution. The solution concentration is chosen to be higher than that of the environment so that a pulse can be seen in the sensor data as the solution is delivered and then absorbed into the environment. Significant changes in the amplitude or slope of this resulting pulse across spikes could be used as an indication that the sensor is drifting and should be re-calibrated. Preliminary results are encouraging.

8 Conclusions

Human-assisted sensing falls on a spectrum, where autonomous ENS deployments [32, 35, 36] on one end are designed to operate with minimal or no human intervention, and participatory sensing deployments [6] on the other end require humans as the vehicle for data collection. Assisted sensing systems are in the middle of this spectrum, and consist of packaging, hardware and software components, and deployment algorithms that are designed to engage humans when necessary, but operate autonomously when possible. Some systems engage humans when the network begins to fail [27, 34], Suelo goes one step further by engaging a human when sensors begin to fail.

We explicitly design our system to have a human in the loop for manual data verification and filtering, sensor re-calibration, and replacing of broken sensors. Instead of trying to build a system that can automatically identify and fix faulty data, our system notifies a human when verification or repairs are needed. It also learns to make better suggestions based on feedback from the user. The Suelo Fault Detection System uses two simple mechanisms to learn parameters in exploratory environments: the EWMA incorporates the sensor characteristics of the deployment into the distance threshold that separates valid and faulty data; the labels for regions can be updated by users at any point to teach SFDS to identify new behavior. Both operations are extremely low overhead, making it possible to update the system parameters as often as necessary.

9 Acknowledgments

This material is based in part upon work supported by the National Science Foundation under Grant numbers 0435497 and CCR-0120778. We would like to thank the anonymous reviewers and shepherd for their comments. Our deployments would not have been possible without the help of Borhan Badruzzaman, Charlie Harvey, John Hicks, Jenny Jay, Christine Lee, Tiffany Lin, and Sarah Rothenberg.

10 References

[1] S. Bapat, V. Kulathumani, and A. Arora. Reliable estimation of influence fields for classification and tracking in unreliable sensor networks. In *IEEE Reliable Distributed Systems*, 2005.

[2] P. Barham, R, Isaacs, R. Mortier, and D. Narayanan. Magpie: Online modelling and performance-aware systems. In *HotOS*, 2003.

[3] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, 2002.

[4] J. Bertrand-Krajewski, J. Bardin, M. Mourad, and Y. Beranger. Accounting for sensor calibration, data validation, measurement and sampling uncertainties in monitoring urban drainage systems. *Water Science and Technology*, pages 95–102, 2003.

[5] D. Bohus, B. Langner, A. Raux, A. Black, M. Eskenazi, and A. Rudnicky. Online supervised learning of non-understanding recovery policies. In *IEEE Spoken Language Technology Workshop*, 2006.

[6] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. In *Procs. of Sensys, Worldwide Sensor Web Workshop*, 2006.

[7] R. Dara, S. Kremer, and D. Stacey. Clustering unlabeled data with soms improves classification of labeled real-world data. In *Intl. Joint Conf. on Neural Networks*, 2002.

[8] A. Demiriz, K. Bennett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. In *Artificial Neural Networks in Eng.*, 1999.

[9] J. Feng, S. Megerian, and M. Potkonjak. Model-based calibration for sensor networks. In *IEEE Intl. Conf. on Sensors*, October 2003.

[10] A. Fox, E. Kiciman, D. Patterson, M. Jordan, and R. Katz. Combining statistical monitoring and predictable recovery for self-management. In *Workshop on Self-Managed Systems*, October 2004.

[11] S. Ganeriwal, L. Balzano, and M. B. Srivastava. Reputation-based framework for high integrity sensor networks. *ACM TOSN.*, 4(3), 2008.

[12] J. Gupchup, A. Sharma, A. Terzis, R. Burns, and A. Szalay. The perils of detecting measurement faults in environmental monitoring networks. In *DCOSS*, 2008.

[13] T. C. Harmon, R. F.Ambrose, R. M. Gilbert, J. C. Fisher, M. Stealey, and W. J. Kaiser. High-resolution river hydraulic and water quality characterization using rapidly deployable networked infomechanical systems (nims rd). *Environmental Engineering Science*, 24(2):151–159, 2007.

[14] C. Harvey, C. Swartz, A. Badruzzman, N. Keon-Blute, W. Yu, M. Ali, J. Jay, R. Beckie, V. Niedan, D. Brabander, P. Oates, K. Ashfaque, S. Islam, H. Hemond, and M. Ahmed. Arsenic mobility and groundwater extraction in bangladesh. *Science*, 2002.

[15] J. W. Hines and B. Rasmussen. On-line sensor calibration verification: A survey. In *Intl. Cong. and Exhibition on Condition Monitoring and Diagnostic Engineering Management*, Sept 2001.

[16] E. Kıcıman and A. Fox. Detecting application-level failures in component-based Internet services. In *IEEE Transactions on Neural Networks*, Spring 2005.

[17] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli. On-line fault detection of sensor measurements. *IEEE Sensors*, Oct 2003.

[18] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *Parallel and Distributed Processing Symposium*, 2006.

[19] L. B. Larkey, L. Bettencourt, and A. Hagberg. In-situ data quality assurance for environmental applications of wireless sensor networks. Technical Report LA-UR-06-1117, Los Alamos Laboratory, 2006.

[20] J. McNeill and V. Winiwarter. Breaking the sod: Humankind, history, and soil : Soils: The final frontier. *Science*, June 2004.

[21] E. Nath and B. Nath. Context-aware sensors. In *EWSN*, 2004.

[22] K. Ni, N. Ramanathan, M. Chehade, L. Balzano, S. Nair, S. Zahedi, G. Pottie, M. Hansen, , and M. Srivastava. Sensor network data faulty types. *Transactions on Sensor Networks*, 2008.

[23] M. L. Polizzotto, B. D. Kocar, S. G. Benner, M. Sampson, and S. Fendorf. Near-surface wetland sediments as a source of arsenic release to ground water in asia. *Nature*, July 2008.

[24] L. J. Puckett, C. Zamora, H. Essaid, J. Wilson, H. M. Johnson, M. Brayton, and J. R. Vogel. Transport and fate of nitrate at the ground-water/surface-water interface. *Environmental Quality*, 37(3):1034–1050, 2008.

[25] N. Ramanathan, L. Balzano, M. Burt, T. Harmon, C. Harvey, J. Jay, E. Kohler, S. Rothenberg, M. Srivastava, and D. Estrin. Rapid deployment with confidence: Calibration and fault detection in environmental sensor networks. In *CENS Tech Report #62*, 2006.

[26] N. Ramanathan, L. Balzano, D. Estrin, M. Hansen, T. Harmon, J.Jay, W. Kaiser, and G. Sukhatme. Designing wireless sensor networks as a shared resource for sustainable development. In *Procs. of Intl. Conf. on Information and Communication Technologies and Development*, 2006.

[27] N. Ramanathan, K. Chang, R. Kapur, L. Girod, E. Kohler, and D. Estrin. Sympathy for the sensor network debugger. In *SenSys*, 2005.

[28] N. Ramanathan, T. Schoellhammer, E. Kohler, and D. Estrin. Fixing faults in wireless sensing systems. Technical report, Center for Embedded Networked Sensing, 2008.

[29] J. Y. Ramel, S. Leriche, M. L. Demonet, and S. Busson. User-driven page layout analysis of historical printed books. *Intl. J. on Document Analysis and Recognition*, 9(2), 2007.

[30] C. C. Rundle. A beginners guide to ion-selective electrode measurements, 2005.

[31] D. Scholefield, T. L. Goff, J. Braven, L. Ebdon, T. L. T, and M. Butler. Concerted diurnal patterns in riverine nutrient concentrations and physical conditions. *Science of the total environment*, May 2005.

[32] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *Procs. Sensys*, 2004.

[33] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler. Lessons from a sensor network expedition. In *Proc. EWSN*, January 2004.

[34] G. Tolle and D. Culler. SNMS: Application-cooperative management for wireless sensor networks. In *Procs. SenSys*, 2004.

[35] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Proc. SenSys*, Nov. 2005.

[36] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *Procs. of EWSN*, 2005.

[37] K. Whitehouse and D. Culler. Macro-calibration in sensor/actuator networks. *Mobile Networks and Applications Journal (MONET)*, June 2003.

[38] Y. Xie. Hcss: Designing reliable embedded systems atop of unreliab platform.

[39] W. Yu. Socio-hydrologic approaches for managing groundwater contamination problems: strategies for the arsenic problem in bangladesh. doctoral thesis, division of engineering and applied sciences. 2003.