Teaching Old Sensors New Tricks: Archetypes of Intelligence

Dimosthenis Karatzas, Arsenia Chorti, Neil M. White, Christopher J. Harris

Abstract— **In this paper a generic intelligent sensor software architecture is described which builds upon the basic requirements of related industry standards (IEEE 1451 and SEVA BS-7986). It incorporates specific functionalities such as real-time fault detection, drift compensation, adaptation to environmental changes and autonomous reconfiguration. The modular based structure of the intelligent sensor architecture provides enhanced flexibility in regard to the choice of specific algorithmic realizations. In this context, the particular aspects of fault detection and drift estimation are discussed. A mixed indicative/corrective fault detection approach is proposed while it is demonstrated that reversible/irreversible state dependent drift can be estimated using generic algorithms such as the EKF or on-line density estimators. Finally, a parsimonious density estimator is presented and validated through simulated and real data for use in an operating regime dependent fault detection framework.**

Index Terms— **intelligent sensor, software architecture, drift estimation, fault detection, data fusion, density estimation, adaptability, reliability, calibration**

I. Introduction

THE TERM "intelligent" when used to qualify a sensor has at best a dubious meaning. Intelligent or smart sensors traditionally refer to sensors offering additional functionality provided by the integration of microprocessors, microcontrollers or application-specific integrated circuits (ASICs) within the sensing element itself. In any case, the exact functionality that would qualify a sensor as intelligent is not as yet strictly defined. As a result, there are many flavors of sensor intelligence available in the literature, ranging from implementations including sensors coupled with application dependent electronics [1], [2], [3], to Self-Validating (SEVA) sensors [4], [5] which place an emphasis on their ability to validate their output. In parallel, the issue of intelligent sensors has been approached from a functional point of view in [6] and [7]. This work concentrates not so much on *how* data is locally processed but on *which* type or results should be provided by an intelligent sensor in terms of available services.

To address intelligent sensors, the IEEE introduced the IEEE 1451 family of standards [8], [9], [10], [11]. IEEE 1451 provides a formal definition of the basic requirements of a smart transducer. These standards define the fundamental element of intelligence as the ability to self-identify, and complement this with on board memory and a set of digital, analogue and mixed communication interfaces. IEEE 1451 has been received well by industry, with a number of compliant products and implementations [12], [13].

While IEEE's intelligent sensor provides basic functionality, the SEVA sensors approach addresses the issue of intelligence at a higher level suggesting that an intelligent sensor should be able to validate its output and communicate sensor state information to a higher level. The SEVA approach recently became a British Standard (BS-7986) [14] which has already been endorsed by parts of industry. BS-7986 defines a set of state values and couples these with rules for the propagation of the validated output and uncertainty values through any modules comprising the intelligent sensor.

It is important to note two key facts, BS-7986 avoids the need to specify how the validated values and uncertainties are actually calculated or when specific sensor states should be entered into, since this functionality is deemed to be application-specific. In addition, it does not make any assumptions regarding the communication interface nor it provides any standard way to identify the sensor. On the other hand, the IEEE 1451 standard defines this lower level functionality but avoids stepping into the realm of higher level data processing. As such the two standards are essentially complementary, but even so they still fail to propose a specific framework for higher-level on-board data processing.

This paper fills this gap by addressing a multitude of common situations encountered in sensory applications through a generic software architecture for intelligent sensors. It is argued that an intelligent sensor should be able to perform on board signal processing within the sensor's software in order to produce the optimal output signal in a variety of adverse circumstances. The software architecture proposed provides a generic framework to implement BS-7986, whilst building upon the requirements introduced by IEEE 1451 to incorporate the following functionality:

- Real-time fault and drift detection, fault isolation and signal conditioning.
- Communication of sensor condition to the sensor management level.
- Adaptation to environmental changes.
- Autonomous reconfiguration (where possible) to continue effective sensor operation despite sensor degradation.

The key contribution of this paper is the presentation of the intelligent sensor software architecture. The architecture comprises modules which address common issues of real world applications such as fault detection and drift estimation. Through a modular approach, the architecture can be easily adapted to fit specific applications, while retaining compatibility with industry standards. In addition to presenting the

Manuscript received June 15, 2006;

D. Karatzas, A. Chorti, N.M. White and C.J. Harris are with the Electronic Systems Design Group, Department of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK (e-mail: dk3@ecs.soton.ac.uk, ersi.chorti@gmail.com, nmw@ecs.soton.ac.uk, cjh@ecs.soton.ac.uk)

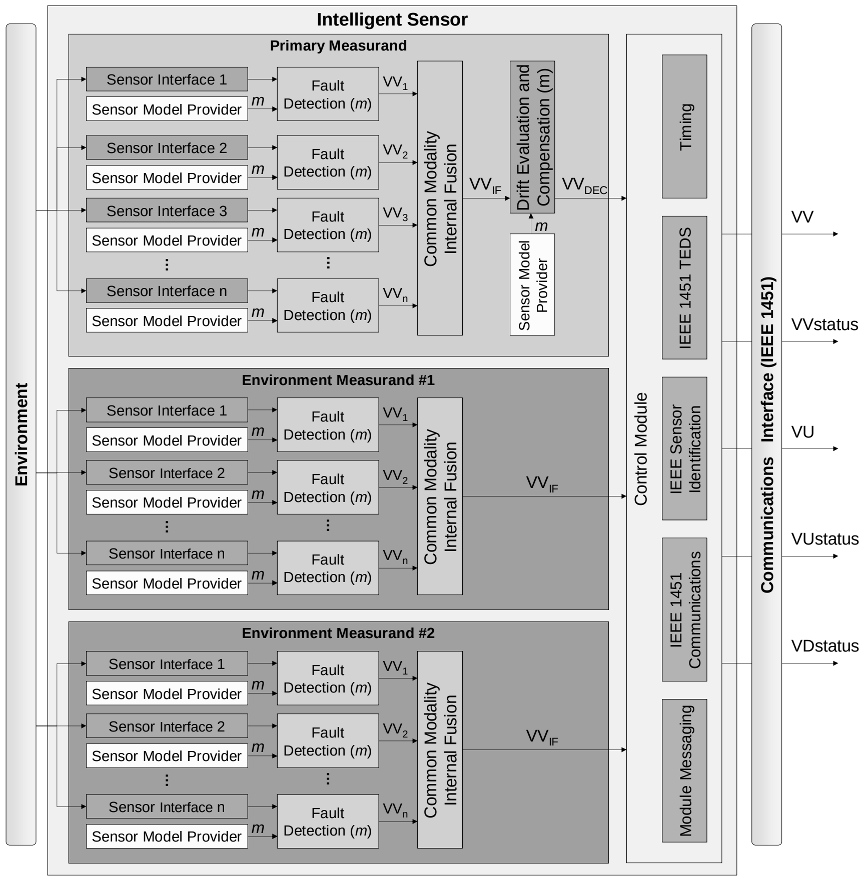


Fig. 1. Intelligent sensor software architecture

intelligent sensor software architecture, we discuss in detail possible algorithmic approaches for the realization of the key components: fault detection, drift estimation and sensor modeling.

The paper is organized as follows: an overview of the intelligent sensor software architecture is given in section II, along with a detailed description of the comprising software modules in subsections II-A to II-F. Section III offers a discussion on existing approaches to fault detection and proposes an algorithm based on an Adaptive Kalman Filter (AKF) enhanced with outlier identification capabilities. In section IV, we analyze possible approaches for drift estimation and demonstrate the identification of reversible state dependent drift using the Extended Kalman Filter (EKF) algorithm. We examine the possibility of using on-line density estimators for irreversible state dependent drift estimation and present a theoretical analysis for multiplicative drift in sensors that include internal oscillatory systems as a result of phase noise. Finally, in section V we demonstrate the use of a sparse density estimator, while section VI concludes the paper.

II. Intelligent Sensor Software Architecture

We consider an intelligent sensor to be a system of primary sensing elements and associated software modules acting as a single entity. The software architecture addresses the following issues:

- Specifies the functionality of the software modules.
- Defines how the modules are interconnected.
- Describes how data propagate between the modules.
- Explains how additional, specialized software modules can be incorporated to tackle application specific tasks.

The software architecture supports input from multiple sensing elements in different configurations, promoting redundancy of information through the use of sensor arrays for each of the quantities being measured. An overview of the modular structure is presented in Fig. 1 to demonstrate the key elements of the architecture. These are explained in more detail in subsequent subsections.

In the configuration of Fig. 1, the quantity of interest (primary measurand) is measured by an array of sensors. Each sensor is interfaced with the software architecture through a Sensor Interface (SI) module (subsection II-A), which is responsible for communicating with the hardware and performing basic signal conditioning. The output of each SI module is individually monitored by a Fault Detection (FD) module (subsection II-C), which assesses each new measurement to produce an associated uncertainty value. The outputs of the FD modules in the array are subsequently combined by an Internal Fusion (IF) module (subsection II-D) and the single value and associated uncertainty is fed to the Drift Estimation and Compensation (DEC) module (subsection II-E), which monitors the behavior of the signal over longer timescale in order to identify and compensate for drift or bias.

Both the FD and the DEC modules generally use additional information about the sensor in terms of precalculated sensor models. A bank of such models is provided by the Sensor Model Provider (SMP) module (subsection II-B). Depending on the algorithm employed by the FD or the DEC module, the theoretical model could refer to different types of sensor models. Particular applications of a Kalman Filter (KF) based FD module and an Extended Kalman Filter (EKF) based DEC module will be described later on in sections III and IV respectively.

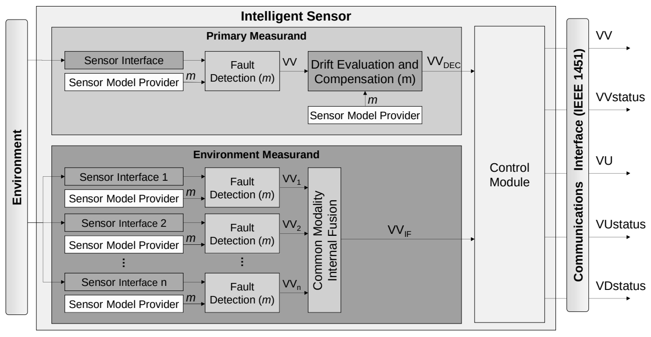
In addition, it is possible that the selection of the theoretical model is also dependent on the particular environmental conditions (regime) within which the intelligent sensor operates. Generally, the environment can be monitored by one or more arrays of sensors (two in the scenario of Fig. 1) measuring various environmental attributes. Modules such as the SMP can use this information to facilitate the selection of the appropriate theoretical model at any given time. A parsimonious density estimator is proposed in section V as a viable solution for the calculation of pdf models. Each attribute monitored by the intelligent sensor is implemented by a branch of modules. The Intelligent Sensor Control (ISC) module (subsection II-F) is the final recipient of all information from the module branches and is responsible to communicate the final sensor output to higher level processes through an IEEE 1451 compliant communications interface.

There are various advantages of having a modular software architecture. Each module is self-contained; it is therefore trivial to exchange a specific module with another of the same functionality without affecting the overall design. The architecture specifies the functionality and the interconnections for each module type, but does not dictate the use of a specific algorithm to achieve the desired effect. The exact algorithm used, is application-dependent and the end user will be able to either choose from a library of alternatives, or produce an application specific implementation. Another advantage of the software architecture is that it allows modules to be combined in a number of alternative ways. Two possible implementations of the architecture are shown in Fig. 2 for illustration purposes.

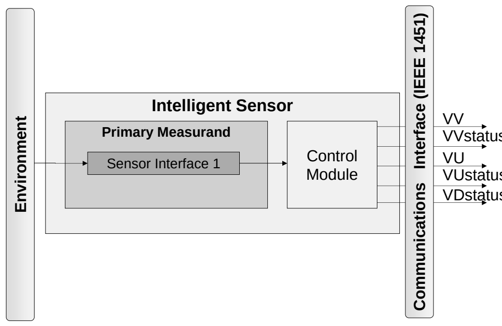
In terms of data propagation, the software architecture is fully compliant with standard BS-7986. All software modules in the data pipeline communicate information using a data structure comprising five pieces of information as required by BS-7986:

i Validated value (VV). This is the measurement possibly after certain corrections have been applied to it.
ii Validated value status (VV_{status}). A byte (8-bit) value indicating how the associated VV data has been generated.
iii Validated uncertainty (VU). Error band of the associated VV data value at 95% level of confidence represented in the same units and with the same precision as the VV.
iv Validated uncertainty status (VU_{status}). A byte value indicating how the associated VU data has been generated.
v Validated device status (VD_{status}). A byte describing the status of the module reporting the above values (e.g. maintenance information and hardware problems).

The intelligent sensor software architecture implements the transition decision rules for VU_{status} and VD_{status}, as well as the rules for deriving VV_{status} as specified in the standard. In general, the architecture assumes VV and VU to be vectors of values. Software modules can be viewed as BS-7986 function blocks with the exception of the SMP module, which provides information outside the main data-pipeline. In the case of the SMP module, the output refers to a theoretical sensor model and the uncertainty value to the probability that this model is valid for the current regime.



(a)



(b)

Fig. 2. Possible implementations of the intelligent sensor architecture

BS-7986 function blocks are defined as software functional units. The function block output value (VV_{out}) is derived from the function block input(s) by applying the algorithms implemented in the block by its designer. So in the general case:

$$VV_{out} = f(VV_1, VV_2, \dots, VV_N) \tag{1}$$

where VV_{out} is the function block output at the corresponding sample, $f(\cdot)$ is the function used to derive VV_{out} from the inputs to the function block and $VV_i, i = 1 \dots N$ is the ith input to the function block. The function block output uncertainty value is then derived from the inputs and their uncertainties by applying (2):

$$U(VV_{out})^2 = \sum_{i=1}^{N} \left[\frac{\partial f}{\partial VV_i} \right]^2 U(VV_i)^2 \tag{2}$$

where $U(VV_{out})$ is the uncertainty associated with VV_{out} at the corresponding sample time, $U(VV_i)$ is the uncertainty associated with VV_i and $\frac{\partial f}{\partial VV_i}$ is the sensitivity coefficient describing how VV_{out} varies with changes in VV_i. The intelligent sensor software architecture modules maintain compatibility with the above notion of function blocks, with the exception of the SMP as explained above.

An additional feature of the architecture is support for communication between modules which is achieved through a messaging mechanism. Although messaging is supported by all modules by default, its use is not necessary for implementing the basic functionality of the architecture. Nevertheless, it is important for such a mechanism to exist, in order to implement more advanced functionality, e.g. in the context of regime change detection. The types of software modules that comprise the intelligent sensor software architecture are described in detail in subsections II-A to II-F. The functionality of each module is defined, along with the standard implementation and possible implementation alternatives where applicable.

A. *Sensor Interface Module (SI)*

The SI module is responsible for interfacing the sensing element to the software architecture. As such, the possible implementations of this module are as many and varied as the sensing elements available. In order for a particular sensing element to be supported by the architecture the single requirement is that a SI implementation is provided for it. The basic functionality of the SI module is to:

- Communicate with the sensing element hardware.
- Obtain measurements on demand.
- Perform basic signal processing (linearization, A/D conversion, conversion to engineering units, etc.).

The standard implementation of the SI module reports the sensor's accuracy (known from the technical characteristics) as the uncertainty value (VU). Subsequent modules (such as FDs) can ignore this information and produce their own. Nevertheless, in the absence of such subsequent modules, this is the best uncertainty that can be associated with the measurement at this level.

B. *Sensor Model Provider Module (SMP)*

The SMP module acts as a repository of pre-calculated sensor models. The basic functionality of the SMP module is to store and communicate pre-calculated models for a specific sensor (or range of sensors), given a model type.

Although a theoretical sensor model is sensor-specific information, it has been a design decision not to include this information in the SI module for two reasons. First, it is a common situation to have an array of same sensors (which share the same models) in the design. Including sensor model information in each instance of the SI would be superfluous. Instead a single SMP module can serve any number of subsequent modules. Second, there are cases where the sensor models refer to a virtual sensor[1] as opposed to a real one. An example of such a situation will be discussed later on in the context of drift estimation and correction. In such situations an SMP module can still act as a model repository.

[1]The "virtual sensor" refers to the theoretical system that has the same macroscopic properties and characteristics with the array/set of the real sensors.

The definition of a sensor model is intentionally left open to mean any set of information that can describe the operation of the sensor. Three model types are already defined:

i Linear State-Space models (intended to be used with KF based implementations of software modules).

ii Non-linear State-Space models (intended to be used with EKF based implementations of software modules).

iii pdf models (intended to be used with probabilistic hypothesis based implementations of software modules).

Models are nevertheless not restricted to the above types, and users can enrich the list of model types with their own. The selection of a specific model in the standard implementation is based on a single attribute; the model type. The standard implementation does not dictate a specific model selection algorithm in the case where more than one model is available. Derived implementations, of course, can make use of additional information and elaborate algorithms to facilitate model selection. Such extra information can be, for example, environmental regime data as discussed in section V.

C. *Fault Detection Module (FD)*

The FD module is responsible to assess and correct the incoming data and produce an uncertainty value. The basic functionality of the FD module is defined as:

- Produce an uncertainty value for each input data value.
- Correct incoming data if possible.

Compliant to the BS-7986 standard, the uncertainty value denotes the error band of the associated data value at 95% level of confidence and is represented in the same units and with the same precision as the data value. The uncertainty value is typically calculated by assessing the new data value in the context of recent data history. Specific algorithms for achieving that are discussed later on in section III.

D. *Internal Fusion Module (IF)*

The IF module enables the configuration of sensors into arrays. It combines an arbitrary number of input data values and associated uncertainties to produce a single sensory output and uncertainty value. Additionally, the IF module is able to identify cases when an input value is inconsistent, and take action to report the problem. The functionality of the IF module is summarized to the following:

- Generation of a single value and associated uncertainty from multiple inputs.
- Filtering of inconsistent values.

The standard implementation of the IF module is based on the Optimal Weight Measurement Fusion (OWMF) algorithm [15]. Let VV_i be the ith input value of the IF module and VU_i its uncertainty value at 95% level of confidence. Assuming that the input values are drawn from the Gaussian distributions $N(VV_i, \sigma_i^2)$, where $2\sigma_i = VU_i$, a typical method to perform data fusion is by combining the inputs based on a minimum mean square error criterion, where the combined output VV_{out} is computed as follows:

$$VV_{out} = f(VV_1, VV_2, \ldots, VV_n) = \frac{\sum_{i=1}^{N} \sigma_i^{-2} VV_i}{\sum_{i=1}^{N} \sigma_i^{-2}} \quad (3)$$

The variance associated with the combined values is given by:

$$\sigma_{out}^2 = \left(\sum_{i=1}^{N} \sigma_i^{-2} \right)^{-1} \quad (4)$$

(4) is the solution to (2) with $f(\cdot)$ given by (3). As such the OWMF algorithm satisfies the BS-7986 requirements for data propagation through function blocks as detailed in the previous section.

Although in most practical cases, the original measurements are drawn from normal distributions this assumption may not always stand. In such cases an application specific implementation of the IF module can be provided. In addition, alternative implementations may perform advanced cross-validation of input measurements and perform fault detection at the sensor array level. Other alternatives for IF include an approach to consistency checking and data fusion between SEVA sensors proposed in [16].

E. Drift Estimation and Compensation Module (DEC)

The DEC module, aims to detect drift mechanisms based on the data, and subsequently correct for the introduced measurement bias. This exercise aims to elongate the useful life span of the intelligent sensor and provide advanced maintenance information to higher level processes. The basic functionality of the DEC module is therefore summarized as:

- Estimate the drift (or bias) from historical data.
- Correct the input for drift and update the uncertainty value accordingly.

Most of the algorithms for drift estimation and compensation tend to be application specific so the standard implementation acts as a placeholder and simply propagates the received data. Specific algorithms and implementations to address particular types of drift are discussed in section IV.

The DEC module, similarly to the FD module may make use of additional information about the sensor in the form of a theoretical sensor model.

F. Intelligent Sensor Control Module (ISC)

The ISC module is responsible for managing the rest of the software modules in the architecture while it acts as the gateway to the world. As such, it is responsible for maintaining compatibility with industry standards. The basic functionality of the ISC module is defined as:

- Communicate with higher level processes (IEEE 1451 interface).
- Manage the software modules (i.e. messaging, timing).

The standard implementation of the ISC module implements a STIM (Smart Transducer Interface Module) as per IEEE 1451.2 [9] with a single channel for the primary measurand data of type "buffered data sequence sensor" (IEEE 1451.2), so that internal timing is left with the Intelligent Sensor. Functional addresses open for industry read extensions are used to communicate the rest of the BS standard data to higher processes on demand [8], [9], [10], [11].

The ISC module also takes care of module messaging as a means for modules to exchange information. This can be achieved either by one-to-one communication between modules, or by one-to-many communication, where modules announce information to all interested listeners. Messaging is important for the implementation of advanced functionality (e.g. the selection of a sensor model based on environmental data). In the rest of the paper we discuss algorithmic approaches for the implementation of the fault detection, drift estimation and stochastic modeling of adverse sensory systems.

For completeness, an example realization of the intelligent sensor software architecture in the particular case of a piezoresistive pressure sensor can be found in the Appendix.

III. Fault Detection

The objective of the FD module is to assess incoming data, and produce an uncertainty value to characterize the output. A further desirable characteristic of fault detection, as listed in subsection II-C, is the ability to produce a corrected output if possible. Hereon the former approach, namely the simple assessment of individual measurements, will be referred to as indicative fault detection. The latter approach, involving the correction of input measurements will be referred to as corrective fault detection. Correction here refers to the estimation of the most probable value at the specific time instance given a known history and a new measurement.

There are advantages and disadvantages associated with each approach which stem from their distinct response in particular scenarios. Indicative fault detection is able to identify the existence of outliers in the input data, but there is no mechanism to ensure the continuation of the intelligent sensor's operation after such an event. Indicative fault detection is discussed in more detail in subsection III-A.

The actual performance of corrective fault detection on the other hand is dependant on how much confidence the algorithm places on new measurements versus a-priori knowledge. Assuming higher confidence on a-priori knowledge, results in an algorithm robust to the presence of invalid data, but slow to respond to changes in the measurement statistics. Corrective fault detection is discussed in more detail in subsection III-B.

Ultimately, a fault detection algorithm should be reactive and robust to the presence of invalid data as well as able to provide a corrected output. A combination of the above approaches is desirable, hence a method is presented in section III-C, based on the use of an AKF to provide corrected output values, whilst in parallel it assesses new measurements online in order to identify invalid data.

A. Indicative Fault Detection

In indicative fault detection each incoming measurement is assessed individually. Typically the assessment is achieved

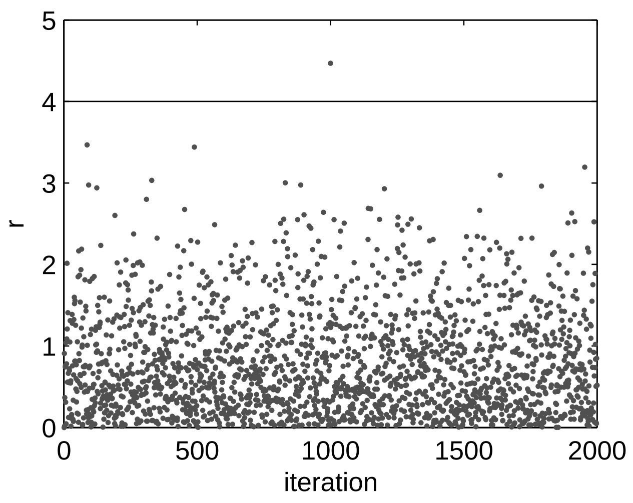


Fig. 3. Normalized KF innovations versus iteration index. A threshold equal to 4 correctly identifies the introduced outlier

by comparing the incoming data to an estimate calculated based on a priori knowledge of the process and previous measurement history.

Assuming a linear stochastic system, residual-based indicative fault detection can be implemented based on a KF. In this case the innovation E produced by the KF is used to assess new data. The magnitude of the inconsistency can be assessed in relation to the known a-priori observation R and state Q noise covariance matrices (for more information on system identification see [17]). The weighted innovation is given by:

$$r = \frac{E}{\sigma}, \text{ where } \sigma = \sqrt{R + CQC^T} \tag{5}$$

where C is the observation mapping matrix of the KF.

In Fig. 3, the ratio r defined by (5) is plotted against the iteration index for simulated data including an outlier at the 1000th iteration. Since r is given relative to σ, a threshold $T = 4$ would ensure that in normal operation 99.994% of the innovations will fall below the threshold.

B. Corrective Fault Detection

In corrective fault detection, the objective is to calculate the current best estimate given new observations as they arrive. In this section we will discuss the use of an Adaptive Kalman Filter (AKF) [18], [19] for corrective fault detection to illustrate the modus operandi of such algorithms.

For fault detection, we are only interested in estimating the observation noise online, since we can typically assume that the process' characteristics are invariant. The system model considered here for illustration is a linear, discrete, stochastic sequence given by:

$$x_k = Ax_{k-1} + w_k \tag{6}$$
$$y_k = Cx_k + n_k \tag{7}$$

where x is the state vector, A is the state transition matrix, y is the observation vector, C is the observation mapping matrix, w is the state noise and n is the observation noise. The noise terms w and n are assumed white Gaussian noise sequences with covariance matrices Q and R respectively. The state noise is assumed zero-mean, while if systematic errors occur the observation noise mean will be non-zero.

The method to estimate the mean and covariance of the observation noise is based on a limited memory algorithm proposed by Myers and Tapley [20]. An approximation to the observation noise sample r_k is:

$$r_k = y_k - C\hat{x}_k^- \tag{8}$$

where $\hat{x}_k^-$ is the a-priory estimate of x_k at time k. An unbiased estimator for r is the sample mean estimated over N samples. It can then be shown [20] that the unbiased estimate of R is given by:

$$\hat{R} = \frac{1}{N-1}\sum_{i=1}^{N}\left[(r_i - \hat{r})(r_i - \hat{r})^T - \frac{N-1}{N}C\hat{P}_k^- C^T\right] \tag{9}$$

where $\hat{P}_k^-$ is the a priori state covariance estimate at time k.

The selection of the buffer size N implies a trade-off between building a non-responsive filter and one prone to the presence of outliers in the observation sequence. The optimal size of the buffer is problem specific (e.g. Mehra [19]). For illustration purposes, the response of a filter with $A = 1$, $C = 1$ and $Q = 0.4$ to simulated data (a step change of the real R) is shown in Fig. 4 for buffer sizes $N = 500$, $N = 1000$ and $N = 2000$ points.

An alternative approach to corrective fault detection is to base the estimate on the pdf of the process at the current iteration. This entails the ability to update the pdf on-line as new data become available. Existing methods for calculating a pdf based on a data distribution are designed to work off-line and their extension to the on-line domain is difficult. An example of such a method is discussed in section V.

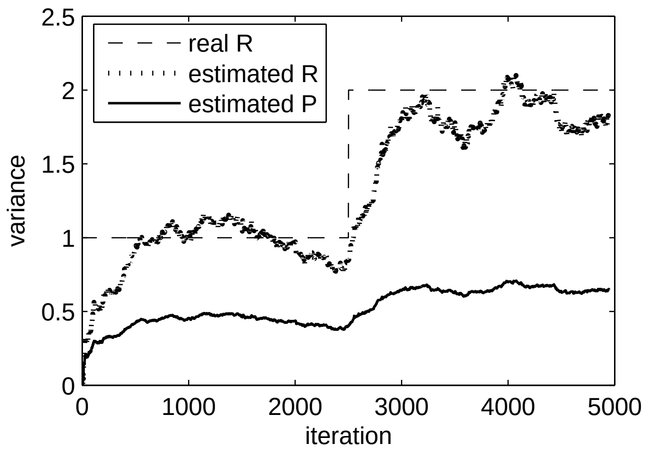
C. Mixed Fault Detection

Let us consider again the use of the AKF as the basis of a fault detection algorithm. A reasonable choice for the AKF buffer size, in the case of the example shown above, would be around $N = 1000$. To examine the response of such a fault detection algorithm to the presence of outliers, an outlier was introduced in simulated data and the filter's response is shown in Fig. 5.

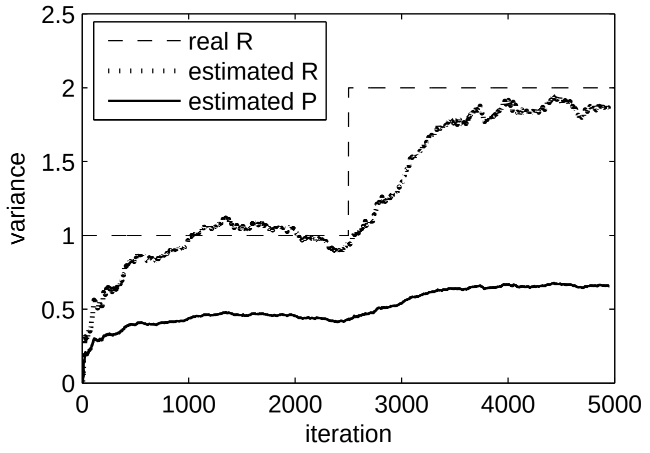
This demonstrates that invalid data [2] may alter the statistics of the AKF buffer dramatically. Moreover, since the data are weighted equally in the sample mean and (9), the effect of invalid data lasts until they exit the buffer. Instead of increasing the buffer size, resulting in a low-responsive fault detection module, it is better to employ an indicative fault detection approach to identify invalid data and prohibit their use in the estimators. A mixed fault detection approach is proposed here as an extension to the basic AKF algorithm.

We propose the use of the algorithm described in subsection III-A for the identification of invalid data. Data points that are identified as invalid are subsequently suppressed in the AKF algorithm, which calculates the estimates of the mean $\hat{r}$ and

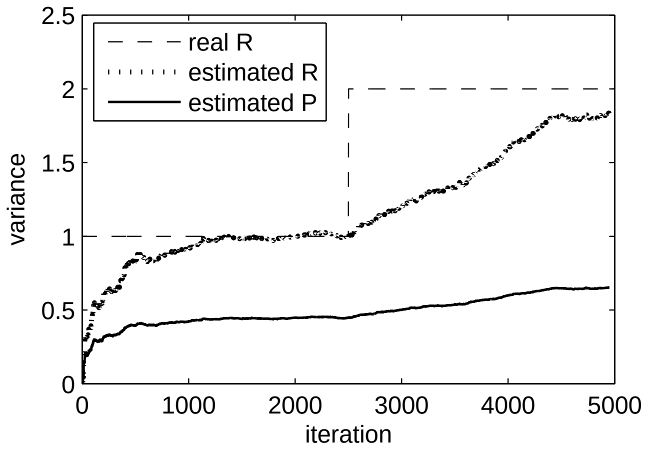
[2]Even a single point, depending on the size of the buffer used.



(a)



(b)



(c)

Fig. 4. AKF algorithm with buffer sizes, from top to bottom, $N = 500$, $N = 1000$, $N = 2000$

covariance $\hat{R}$ of the observation noise based on the remaining data. Alternatively, if an indication of the event is desirable, the invalid data point may be considered momentarily (resulting in a higher estimate of R), but removed from the buffer later on, avoiding an alteration of the AKF buffer statistics for the length of the buffer.

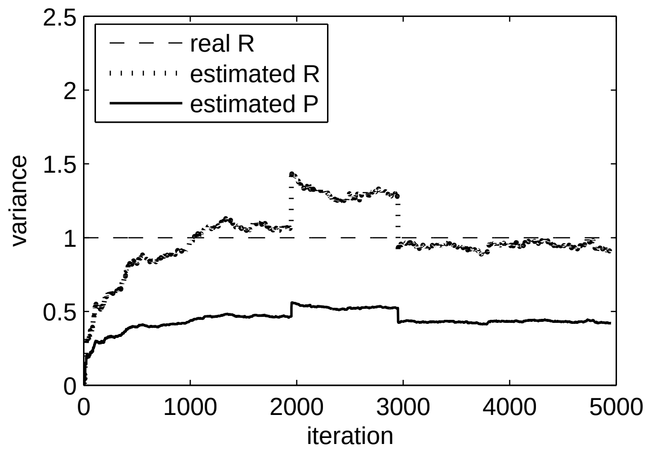


Fig. 5. AKF algorithm response to the presence of an outlier. AKF buffer size $N = 2000$

The algorithm outline is summarized as:

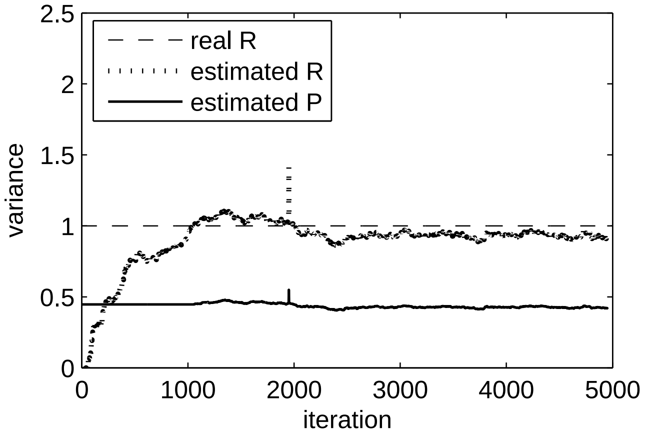
1) At step k, calculate the a priori estimates for $\hat{x}_k^-$ and $\hat{P}_k^-$ based on the standard KF equations. If the measurement at time $k-1$ was identified as invalid and if the estimates used in the KF calculations were derived including this information (see step 4), set $\hat{P}_k^- = \hat{P}_{k-1}^-$.
2) Assess the validity of the new measurement y_k based on its innovation using (5). The covariance R used for this step is the previous estimate $\hat{R}_{k-1}$.
3) Estimate the new values for the mean $\hat{r}_k$ and the variance $\hat{R}_k$ based on the sample mean and (9) taking into account the new data point.
4) If the data point is identified as invalid:
   - If it is desirable to indicate this event to higher level processes, use the estimated observation noise mean $\hat{r}_k$ and variance $\hat{R}_k$ for the rest for the calculations (resulting in a higher reported covariance). If it is desirable to suppress this event use the mean and variance values of iteration $k-1$, $\hat{r}_{k-1}$, $\hat{R}_{k-1}$.
   - Set the observation noise sample r_k to the mean of the valid r_i in the buffer to suppress the use of this data point:

$$r_k = \frac{\sum_{i=k-N}^{k} a_i r_i}{\sum_{i=k-N}^{k} a_i}$$
$$\text{where } a_i = \begin{cases} 1 \text{ if point } i \text{ is valid} \\ 0 \text{ if point } i \text{ is invalid} \end{cases} \tag{10}$$

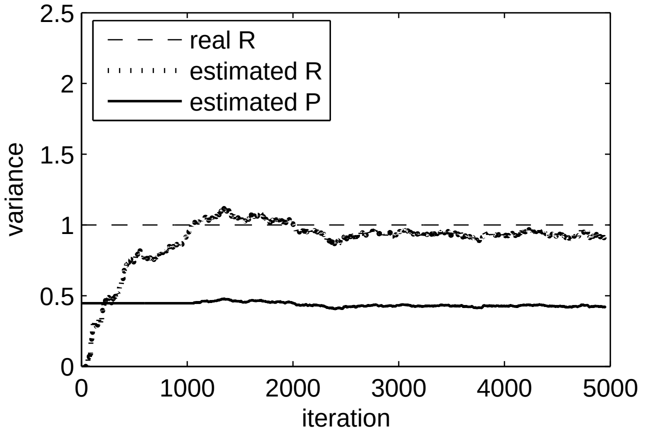
5) Complete this iteration of the KF and go back to step 1.

The response of the fault detection module for the same data used to produce Fig. 5 is shown in Fig. 6. Fig. 6 (a) shows the implementation where the estimates $\hat{r}_k$ and $\hat{R}_k$ are used momentarily to produce an indication of the existence of invalid data and suppressed afterwards. Fig. 6 (b) shows the implementation where invalid data are completely suppressed.

The identification of invalid data enables the fault detection module to eliminate undesirable effects in the AKF buffer statistics without necessitating the use of a larger buffer, retaining therefore the responsiveness of the fault detection



(a)



(b)

Fig. 6. Mixed fault detection with suppression of invalid data

algorithm. The mixed fault detection algorithm described in this section is the default implementation for the FD module of the intelligent sensor architecture and has been employed in the example realization of the intelligent sensor software architecture discussed in the Appendix.

IV. Drift Estimation Algorithmic Approaches

Various uncertain influences (e.g. ageing, environmental conditions, "poisoning" etc.) may generate drift in a sensor's response, introducing the need for self-calibration and self-validation. As previously argued, to avert these effects an integral part of the intelligent sensor is a drift compensation module [16]. A unique generic and universal approach for drift estimation (either additive or multiplicative) cannot, however, be proposed due to the essentially different underlying mechanisms responsible for its generation.

In the literature, specialized approaches have been proposed, e.g. van Putten *et. al.* [21] compare the "chopping" method, the "sensitivity variation" approach and the "geometrical van Putten method" for drift estimation. Commonly, research have been focused on quantitative analyses (physical models) of either the drift generation mechanisms [22], [23] or of the resulting drift itself [24]. We propose a unifying approach for

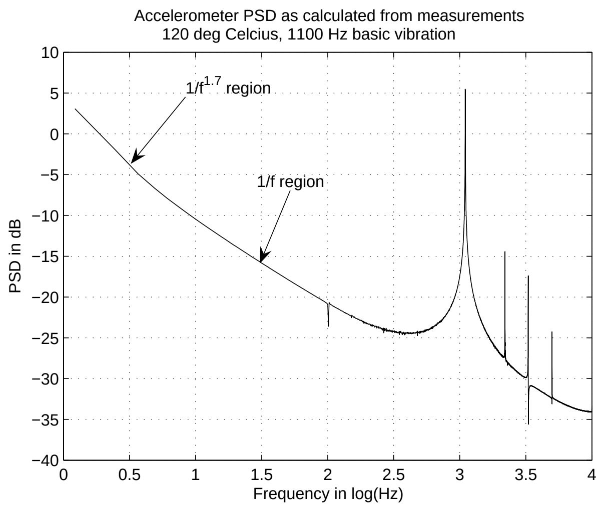


Fig. 7. ADXL203 dual-axis accelerometer power spectral density for a driving frequency of 1100 Hz at a temperature of 120 °C

drift classification inferred from the dynamic and stochastic properties of the output. A central point we introduce in the drift estimation discussion regards the matter of whether drift stems from processes stochastically related to the sensory operation. Furthermore, distinguishing between reversible and irreversible biases, we can classify drift into four major classes:

A. Reversible state dependent drift.
B. Reversible state independent drift.
C. Irreversible state dependent drift.
D. Irreversible state independent drift.

In the following subsections each of these classes is discussed in detail, along with proposed approaches for their identification and compensation in specific cases; e.g. additive drift due to sensor nonlinearities and multiplicative drift due to phase noise of internal oscillators in the sensory system. We have used an Analog Devices ADXL203 dual-axis accelerometer as a case study to test the possible implementations of drift compensation modules for these four classes of drift. The analysis commences with a simple high level model of the accelerometer.

We have performed spectral analyses of measurements and plot in Fig. 7 the Power Spectral Density (PSD) of the x-axis output when the accelerometer was driven by an acoustic vibrator having a sinusoidal input of 1100 Hz. From the sensor PSD we obtain a qualitative representation of the accelerometer/vibrator system as a weakly nonlinear system of order four with the second, third and fourth harmonics being clearly distinguishable in the graph. Furthermore, by simple inspection of the PSD at low frequencies we can identify a $1/f$ type power-law process. This kind of low-frequency processes are considered to arise due to intrinsic electronic noise sources in the accelerometer circuit. Finally, the spectral broadening around the vibrating frequency of 1100 Hz and its harmonics due to phase noise can be modeled as a multiplicative gain drift as stems from theoretical analyses in [25]. A more detailed analysis of this effect is presented in subsection IV-D.

TABLE I
ACCELEROMETER WEAK NONLINEARITY COEFFICIENTS

Polynomial coefficient (Pc)	Pc mean value	Pc variance
a_0	0.0503	$2.0419 \cdot 10^{-8}$
a_1	0.1411	$1.076 \cdot 10^{-6}$
a_2	-0.0023	$1.1431 \cdot 10^{-8}$

Based on the previous remarks, we can model the accelerometer following the input/output relation:

$$y(t) = d(t) + n(t) + b(t) \sum_{k=0}^{4} a_k x(t)^k \tag{11}$$

where $d(t)$ represents additive $1/f$ type noise sources, $n(t)$ is a zero-mean white Gaussian process, $b(t)$ is the multiplicative gain drift due to phase noise, a_k is the kth order coefficient in the Taylor (or Volterra) series expansion of the system time domain response. $x(t)$ is the measurand process, commonly modeled as a Gaussian random process.

A. Reversible, state dependent drift

Reversible, state dependent drift results from non-idealities in the sensor response (the most common example being presented by nonlinearities). In the case where a weak-stationarity requirement is fulfilled, it is possible to develop approximate state space models of such sensors using the EKF [26]. In that aspect, a tangible estimation of the mean drift/offset in the sensor output as a function of the internal state moments is possible.

Of greatest practical interest is when the sensor is a nonlinear device. A variety of sensors have been described by a nonlinear model [27], and various approaches have been used for the circumvention of such natural phenomena [28]. In order to demonstrate the alternative use of the EKF in such sensory systems, we proceed by considering the specific case where the sensor output can be described by a low-order polynomial, representing the sensor as a weakly nonlinear system.

For the derivation of an approximate 2nd order polynomial model for the ADXL203 dual-axis accelerometer during the design of the EKF, we have measured the sensor response for a driving frequency of 100 Hz and obtained $N = 30$ sets of $n = 5000$ measurements. We have used 20 of the sets to obtain a unique averaged 2nd order model for the accelerometer, while using these parameters we algorithmically estimated the drift in the remaining 10 sets. The estimated coefficients of the 2nd order polynomial are included in Table I.

The algorithmic estimation of the drift due to the second order nonlinearity is based on the subsequent equations:

$$\begin{aligned} x_k &= x_{k-1} + w_k && (12)\\ y_k &= a_0 + a_1 x_k + a_2 x_k^2 + n_k && (13)\\ \sigma_x^2 &\simeq \overline{\sigma_{\hat{x}_k}^2} = \frac{(k-1)\overline{\sigma_{\hat{x}_{k-1}}^2} + \hat{x}_k^2}{k} && (14)\\ \overline{dc_k} &= \frac{(k-1)\overline{dc_{k-1}} + a_2 \overline{\sigma_{\hat{x}_k}^2}}{k} && (15)\\ & \quad \text{Overall drift} && \\ dc &= \overline{dc_k} + a_0 && (16) \end{aligned}$$

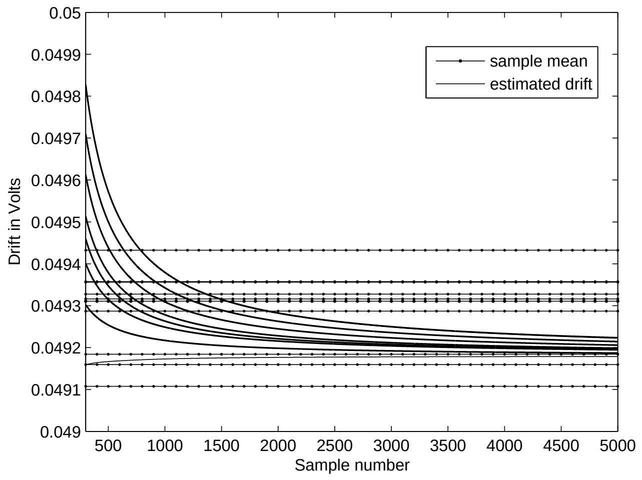


Fig. 8. Estimation of state dependent drift of an accelerometer. The solid lines represent estimations while the dot-lines are calculated as the set empirical offsets

where variances of the white zero-mean Gaussian processes w and n are $Q = 0.0063$ and $R = 10^{-12}$ respectively. Assuming that x is a Gaussian random process, the above expressions converge to the actual variance and mean respectively. (14) and (15) represent the mean values of the variance and dc offset respectively based on the EKF a-posteriori estimates $\hat{x}_k$, while (16) accounts for the overall drift taking into account the polynomial coefficient a_0.

Fig. 8 compares the estimated and empirical (calculated as the mean of the sample set) drift values for the 10 test sets of measurements. The mean estimated drift is 49.2 mV, less than 1% different from the mean empirical drift. Fig. 8 also shows the rapid convergence of the proposed algorithm, with approximately 1000 samples being necessary for a convergence better than 5% error. The proposed algorithm is employed in the example realization of the intelligent sensor in the case of a piezoresistive pressure sensor included in the Appendix.

The algorithmic drift estimation discussed in the present section is not conceptually confined to the case of nonlinearities, but encompasses the whole class of state dependent drift. The non-ideal sensor time-domain response $h(\cdot)$ along with $H(\cdot)$, the Jacobian matrix of its partial derivatives (with respect to x), have to be suitably modified. As an example, the additive drift in the accelerometer x-axis due to a linear cross-correlation to the y-axis can be evaluated through the following simple modifications; a coupled EKF should be constructed for the two accelerometer axes based on relations of the type:

$$\begin{aligned} h_x(x,y) &= a_1 x + \lambda_{xy} y && (17)\\ H_x(x,y) &= a_1 && (18)\\ h_y(x,y) &= \beta_1 y + \lambda_{yx} x && (19)\\ H_x(x,y) &= \beta_1 && (20) \end{aligned}$$

In (17) λ_{xy} represents the cross-correlation of the x-axis to the y-axis, while λ_{yx} in (19) represents the cross-correlation of the y-axis to the x-axis and a_1, b_1 represent linear gains in the two axes.

B. Reversible, state independent drift

Reversible state independent drift can generally be evaluated. A representative example of this class of drift is the gravitational offset (g-offset) in an accelerometer, generated from its relative to the ground angle. A commonly-encountered approach for the accelerometer calibration is through the use of a pair of accelerometers at known relative positions. Alternatively, in the case where the dc component of the sensor output is of no interest, ac coupling or the use of differential circuits in the output is an effective low-cost method for its elimination.

C. Irreversible, state dependent drift

Drift due to irreversible state dependent phenomena is the result of sensor non-stationarity (dynamic behavior) in the time interval of interest[3]. Such drift can severely compromise the performance of a sensory based system, such as an electronic nose. It generally follows a specific trend [29] and various approaches for its compensation have been reported in literature [30] (multiplicative step drift), [31] (additive dynamical drift).

On-line density estimation algorithms specifically designed for systems that are governed by rapid dynamics are currently being researched. In [32] a novelty detection based approach is outlined using recursive dynamic principal component analysis for gas sensor arrays, while in [33] the use of independent component analysis is demonstrated for non-Gaussian data. Conversely, a density estimation approach for drift compensation could be based on the on-line implementation of the density estimation algorithm discussed in section V.

D. Irreversible, state independent drift

Irreversible state independent drift is due to the long-term mean of intrinsic noise sources (such as $1/f$ noise in electronic sensors). In general it is the most difficult part of drift to identify and compensate. In the following we will present a theoretical result for the multiplicative drift estimation in the case of oscillatory-based systems such as accelerometers, direct frequency output vibrating gyroscopes [34] or surface-acoustic-wave and surface-transverse-wave based gas sensors [35] in the presence of phase noise in the internal oscillatory systems. In the case of the latter, close-to-the-carrier phase noise can be the major limiting factor to the sensor noise and its resolution.

As discussed in detail in [25], real oscillators suffer from phase noise that distorts the system long-term stability. The effect is manifested through a spectral broadening (indicating energy spreading) around the oscillation frequency and is modeled by means of power-law phase noise processes with PSD of the type $k_\alpha f^{-\alpha}$, where $\alpha \in \{0, 1, 2, 3, 4\}$ and f is the frequency in Hz.

In order to quantify the resulting drift effect, we begin with considering the complex valued oscillation:

$$\psi(t) = e^{j(\omega_{osc}t+\phi(t))} \tag{21}$$

[3]Any real system exhibits non-stationary behavior if observed for a sufficiently long time. In that context systems can be modeled as stationary as long as their dynamic behavior is negligible for application concerned purposes.

where $\psi(t)$ is an analytic version of a real oscillator at $\omega_{osc} = 2\pi f_{osc}$ assuming negligible amplitude noise. In [25] the correspondence in terms of mean and variance between a real valued oscillator and its complex valued counterpart (analytical representation) is provided. In (21) we can isolate the effect of the phase noise component as a multiplicative term $b(t)$,

$$b(t) = e^{j\phi(t)} \tag{22}$$

with $\phi(t)$ representing the phase noise process. Modeling $\phi(t)$ as a zero-mean Gaussian random process, we can estimate the expected value of the process $b(t)$ as a function of the variance $\sigma_\phi^2(t)$ of the phase noise process $\phi(t)$:

$$\mathbf{E}[b(t)] = \mathbf{E}[e^{j\phi(t)}] = e^{-\frac{\sigma_\phi^2(t)}{2}} \tag{23}$$

where $\mathbf{E}[\cdot]$ denotes statistical expectation.

(23) expresses the actual multiplicative drift in the oscillator output. As a result, drift in the output of oscillatory systems corrupted by phase noise is expressed as a function of the overall variance of the phase noise process and can be time dependent in the case of non-stationary phase noise[4]. The above approach can be incorporated in the design process of sensors based on oscillators and can be particularly useful in early simulation stages.

V. OFR WITH LOO TEST SCORE AND LOCAL REGULARIZATION SPARSE DENSITY ESTIMATION

Fundamental to fault detection algorithms for complex nonlinear sensors is the ability to generate an efficient pdf for the underlying process, so that a metric of its output measurement uncertainty can be evaluated (and propagated through a fusion process) as well as its utilization in a fault detection algorithm. Nonparametric kernel based approaches provide an efficient framework for the estimation of pdfs of representative sample sets (e.g. of size at least $N = 100$). A number of density estimation methods are available with the Parzen windows [36] being the most widely used and universal approach.

Recently, however, sparse density algorithms have become available, [37], [38], [39], in view of the need for decrease in CPU time and memory size for the representation of a probabilistic system model. In applications related to intelligent sensing, the requirement for sparsity is acute. More importantly though, a parsimonious stochastic representation of a sensory system is in better agreement with intuition about the actual physical processes being modeled. In contrast to the Parzen window where the number of kernels employed to represent the system is equal to the size of the representative sample set (100s), sparse kernel density approaches only use a small number of kernels (< 10). The physical system is expected to have a more "concentrated" behavior.

In the present section, we employ the use of the Orthogonal Forward Regression with Leave-One-Out test score and local regularization (OFR-LOO) algorithm [37] for the construction of a sparse probabilistic model for the sensory system. The main motivation behind using the above algorithm is the

[4]σ_ϕ^2 can be a function of time.

fact that the Leave-One-Out test score, a powerful cross-validation technique in nonparametric modeling, is convex in regard to the model complexity. In the case of kernel-based approaches, model complexity is simply represented by the number of kernels employed. The use of the OFR-LOO as a construction[5] algorithm allows the automatic termination of the model selection procedure.

In the following we will demonstrate and validate the use of the OFR-LOO algorithm through a simulated example and then move to the modeling of an accelerometer at different operation conditions. In this context, the proposed use of the OFR-LOO is for the provision of sparse kernel models in operating conditions identified by the Sensor Model Provider module.

In the simulated example we have considered the case of a piezoresistive pressure sensor. In [27] is argued that such sensors are sensitive to the operating temperature T, with their resistance R being described by an expression of the type:

$$R(T) = R(0)(1 + \alpha T + \beta T^2) \tag{24}$$

with $\alpha, \beta \in \mathbb{R}, \beta < \alpha$. The dependance of the sensor resistance on the temperature as described in (24) introduces the following implications: (i) there exists a temperature dependent dc offset in the sensor output (ii) the actual gain K of the sensory element is temperature dependent. As a result of the above, for the same set of pressure measurands P, the probabilistic model that describes the sensor output has an increasing mean and variance with increasing temperature.

The above effects are demonstrated in the simulated piezoresistive pressure sensor, where the output voltage is modeled as:

$$\begin{aligned} V &= K(T) \cdot P & (25) \\ K(T) &= K_0(1 + \alpha T + \beta T^2) & (26) \end{aligned}$$

In our simulation we have chosen the values $\alpha = 0.001$ and $\beta = 0.0001$ that are of the same order of magnitude with estimated values in [21]. In Fig. 9 we present the stochastic modeling of the simulated piezoresistive pressure sensor, using the OFR-LOO algorithm and compare the results with a Parzen window of $N = 10^6$ samples that represents the actual pdf (the Parzen window converges to the actual pdf when the sample size is infinite). The pressure measurand is modeled as a zero-mean Gaussian process of unit variance and the OFR-LOO pdf is estimated over $N = 100$ samples. The OFR-LOO window length was estimated through cross-validation as $\sigma = 1.08$ while the window length of the Parzen window is calculated as $\sigma = 1$.

The OFR-LOO pdf matches well with the Parzen window pdf in all examined temperatures, thus validating the use of the OFR-LOO for pdf estimation, even when a small number of samples ($N = 100$ in the specific example) are available. In the same graph is also shown the slight drift in the mean value of the sensor output pdf, inferring the possible use of the on-line version of the OFR-LOO for drift estimation. More importantly, it is clearly shown that different stochastic models (e.g. distinguishably different variance) should be employed as a result of variations of the environmental conditions.

[5]We "construct" the model starting from a single kernel and moving to higher order representations following a tree based approach for the search of the optimal overall sub-path at each stage of the construction. The process is terminated when a minimum in the LOO test score is achieved

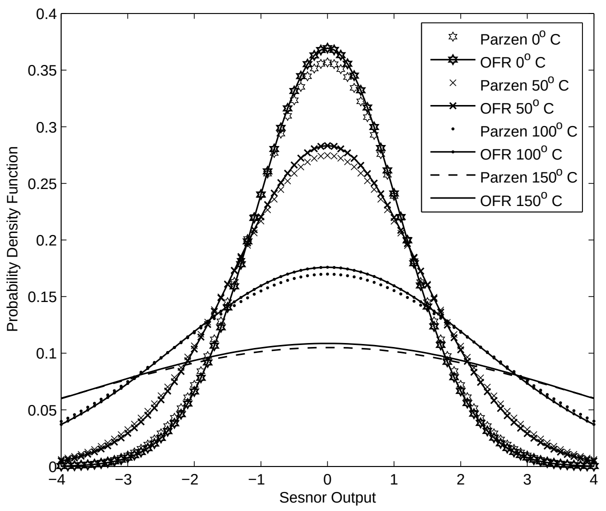


Fig. 9. Comparison of the pdf of a simulated piezoresistive pressure sensor, using the OFR-LOO algorithm with window length $\sigma = 1.08$ and Parzen windows with window length $\sigma = 1$

TABLE II
PDF ESTIMATION FOR THE SIMULATED PIEZORESISTIVE PRESSURE SENSOR

Temperature	Mean number of kernels	Std. in kernel number
0^oC	1.4	0.7818
50^oC	1.43	0.6553
100^oC	1.54	0.7577
150^oC	1.45	0.7961

In Table II we present the results of 100 independent runs of the OFR-LOO algorithm for the simulated piezoresistive pressure sensor and have calculated the mean number of kernels employed in the proposed stochastic modeling along with the standard deviation in the number of kernels. It is worth noting that the stochastic process can be reliably represented by 1 or 2 kernels, in contrast to traditional stochastic models that make use of the totality of the sample set ($N = 100$).

The previous discussion demonstrates that density estimation algorithms can be employed in the realization of novelty detection based fault detection operations. Moreover, on-line parsimonious density estimators are suitable for the identification of irreversible state dependent drift as discussed in subsection IV-D. In the rest of the section we use the OFR-LOO algorithm on real ADXL203 accelerometer data.

The OFR-LOO algorithm has been used for the modeling of the accelerometer in different operating temperature conditions, when driven by an acoustic vibrator at 100 and 200 Hz and have found that the pdfs coincide for these two driving frequencies, inferring that the probabilistic accelerometer model is independent of the measurand harmonics. Furthermore, the accelerometer pdf was estimated for a range of operating

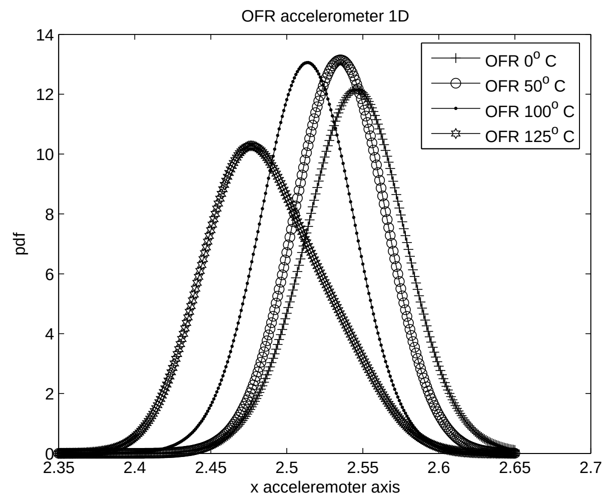


Fig. 10. Estimation of pdf from real accelerometer data using the OFR-LOO algorithm

TABLE III
PDF ESTIMATION FROM REAL ACCELEROMETER DATA

Temperature	Mean number of kernels	Std. in kernel number
0^oC	3.01	0.1
50^oC	2.56	0.4989
100^oC	2.22	0.4399
125^oC	2.98	0.1407

temperatures, 0^oC, 50^oC, 100^oC and 125^oC over $N = 100$ samples with Gaussian kernels of length estimated through cross-validation as $\sigma = 0.0285$. The results are presented in Fig. 10.

Using 100 sets of $N = 100$ measurements, we have repeated the above pdf calculation and have estimated the mean number of kernels and the standard deviation in the number of kernels required from the OFR-LOO algorithm to model the accelerometer in the previously mentioned operation conditions. The results are presented in Table III.

Two main conclusions can be drawn from the above. There appears to be a noticeable drift in the accelerometer output as shown by the mean of the pdf in different temperatures. Although we believe that this is an artefact of the experimental set-up due to accelerometer rotating during the measurements, it is clear that on-line density estimation can provide information about drift. Secondly, the shape of the pdf at 125^oC is distinguishably different from the pdf of the rest of the operating temperatures. The nominal temperature operating range is defined as 0^oC-110^oC and as a result the stochastic modeling employing the OFR-LOO algorithm can separate the "normal" sensor output from the "corrupted" due to operating outside the range of nominal conditions.

The benefit of the approach is that it utilizes sensor data alone, no a-priory physical models are necessary and conventional statistical hypothesis fault condition algorithms can be used directly with the derived pdfs. Such abstract sensor representations capture the underlying physical processes avoiding the development of detailed quantitative models. Also, the approach is particularly amenable to sensors with nonlinearities dependent on the operating regime.

VI. CONCLUSIONS

In this paper we have proposed a generic, modular software architecture as an advantageous intelligent sensor implementation. Such an approach enables on-board signal processing to produce the optimal signal output. The proposed architecture bridges the gap between existing industry standards (IEEE 1451 and BS-7986) by exposing higher level signal processing (BS-7986) while adhering to low-level prerequisites set by IEEE 1451. The software architecture aims at addressing the totality of the intelligent sensor goals, namely optimal data fusion, real-time fault detection, calibration and autonomous reconfiguration.

We have examined a variety of possible algorithmic implementations of the above operations and have proposed a mixed indicative/corrective approach for fault detection. In terms of drift estimation, we have shown that the EKF can be employed in reversible state dependent drift estimation, while on-line density estimation can be utilized when the drift is irreversible state dependent. Furthermore, a state of the art sparse density estimation algorithm was used in a parsimonious model selection context. It was demonstrated that it can be used with a statistical hypothesis algorithm to generate fault condition, even when no physical sensor model is available over a range of operating regimes.

A snapshot of the developed demonstrator of the intelligent sensor software architecture is included in the Appendix, using synthetic data.

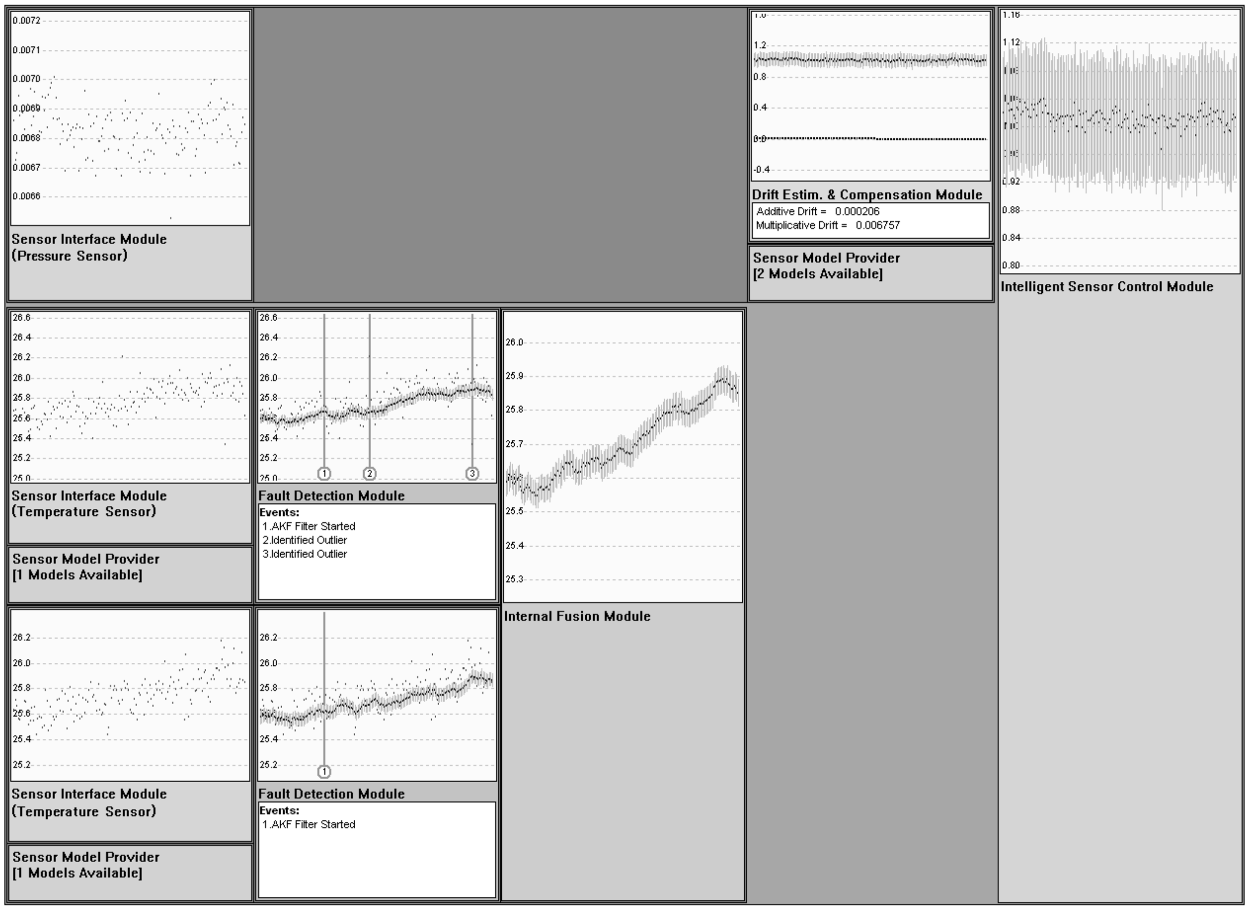
APPENDIX

The example implementation illustrated in Fig. 11 features a piezoresistive pressure sensor as the primary sensing element while temperature sensors are used to monitor the environment and choose the right sensor model for drift compensation during the operation. The current implementation of the intelligent sensor software architecture is developed in software using an object oriented language to facilitate the specialization of modules as necessary. However, as long as the architecture's definition is adhered, all or parts of the modules of the architecture can be implemented in hardware where this offers an advantage.

A drift estimation and compensation module is used for calibration of the pressure sensor output V. The dependance of V on the measurand P and the temperature T is described below:

$$V = K(T)(P + 0.01P^2) \tag{27}$$
$$K(T) = 1.5 \cdot 10^{-4}T + 4.3 \cdot 10^{-6}T^2 \tag{28}$$

As a result, the pressure sensor suffers both from additive drift because of the second order nonlinearity in (27) and from multiplicative gain drift as expressed in (28). The additive drift is evaluated using the EKF algorithm as outlined in subsection IV-A. For the identification of the multiplicative gain drift we use the temperature measurements for a straightforward evaluation based on (28).



Additive Drift = 0.000206
Multiplicative Drift = 0.006757

Events:
1.AKF Filter Started
2.Identified Outlier
3.Identified Outlier

Events:
1.AKF Filter Started

Fig. 11. Demonstrator of the intelligent sensor software architecture featuring a piezoresistive pressure sensor as the primary sensor and temperature sensors as environmental conditions monitoring sensors

As far as the temperature sensors are concerned (two in the specific implementation), a mixed indicative/corrective AKF based approach is used, as discussed in subsection III-C, for fault detection before the internal fusion module. In the snapshot, two outliers are identified in the output of the first temperature sensor and are removed before the temperature measurements are fused.

REFERENCES

[1] M. A. Clapp and R. Etienne-Cummings, “A dual pixel-type array for imaging and motion centroid localization,” *IEEE Sensors Journal*, vol. 2, no. 6, pp. 529–548, Dec. 2002.

[2] L. Kish, J. Solis, W. Marlow, R. V. amd C. Granquist, V. Lantto, J. Smulko, and G. Scmera, “Detecting harmful gases using fluctuation-enhanced sensing with Taguchi sensors,” *IEEE Sensors Journal*, vol. 5, no. 4, pp. 671–676, Aug. 2005.

[3] N. A. Riza, M. A. Arain, and F. Perez, “Harsh environments minimally invasive optical sensor using free-space targeted single-crystal silicon carbide,” *IEEE Sensors Journal*, vol. 6, no. 3, pp. 672–685, June 2006.

[4] J. Yang and D. Clarke, “A self-validating thermo-couple,” *IEEE Transactions on Control Systems Technology*, vol. 5, no. 2, pp. 239–253, Mar. 1997.

[5] M. Henry, “Recent developments in self-validating (SEVA) sensors,” *Sensor Review*, vol. 21, no. 1, pp. 16–22, 2001.

[6] J. P. Cassar, M. Bayart, and M. Staroswiecki, “Hierarchical data validation in control systems using smart actuators and sensors,” in *IFAC Symp. Intelligent Components and Instruments for Control Applications (SICICA 92)*, 1992.

[7] M. Staroswiecki, “Intelligent sensors: A functional view,” *IEEE Trans. on Industrial Informatics*, vol. 1, no. 4, pp. 238–249, Nov. 2005.

[8] IEEE, *IEEE Std 1451.1-1999 Standard for Smart Transducer Interface for Sensors and Actuators - Network Capable Application Processor*, J. C. Edison, Ed. IEEE Standard Association, 2000.

[9] ——, *IEEE Std 1451.2-1997 Standard for Smart Transducer Interface - Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheets (TEDS) Formats*, E. V. El-Kareh, Ed. IEEE Standard Association, 1998.

[10] ——, *IEEE Std 1451.3-2003 Standard for Smart Transducer Interface - Digital Communication and Transducer Electronic Data Sheet (TEDS) Formats for Distributed Multidrop Systems*, L. Eccles, Ed. IEEE Standard Association, 2004.

[11] ——, *IEEE Std 1451.4-2004 Standard for Smart Transducer Interface for Sensor and Actuators - Mixed Mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats*, T. Licht, Ed. IEEE Standard Association, 2004.

[12] A. de Castro, “A system-on-chip for smart sensors,” in *Proc. of the 2002 IEEE International Symposium on Industrial Electronics (ISIE 2002)*, 2002.

[13] J. Schmalzel, F. Figueroa, J. Morris, S. Mandayam, and R. Polikar, “An architecture for intelligent systems based on smart sensors,” *IEEE Transactions on Instrumentation and Measurement*, vol. 54, no. 4, pp. 1612–1616, Aug. 2005.

[14] B. S. B. 7986:2005, *Data Quality Metrics for Industrial Measurement and Control Systems - Specification*. BSi, Apr. 2005.

[15] C. Harris, X. Hong, and Q. Gan, *Adaptive Modelling, Estimation and Fusion from Data*. Springer, 2002.

[16] M. Duta and M. Henry, “The fusion of redundant SEVA measurements,” *IEEE Transactions on Control Systems Technology*, vol. 13, no. 2, pp. 173–184, Mar. 2005.

[17] V. Cherkassky and F. Mulier, *Learning from Data: Concepts, Theory and Methods*. Wiley, 1999.

[18] R. Mehra, "On the identification of variances and Adaptive Kalman Filetring," *IEEE Transaction on Automatic control*, vol. 15, no. 2, pp. 175–184, Apr. 1970.

[19] ——, "Approaches to adaptive filtering," *IEEE Transaction on Automatic Control*, vol. 17, no. 5, pp. 693–998, 1972.

[20] K. Myers and D. Tapley, "Adaptive sequential estimation with unknown noise statistics," *IEEE Transaction on Automatic Control*, vol. 21, no. 4, pp. 520–523, Aug. 1976.

[21] M. van Putten and M. van Putten, "Facing drift: a comparison of three methods," *Sensors and Actuators A Physical*, vol. 90, 2001.

[22] S. Jamasb, S. Collins, and R. Smith, "A physical model for drift in pH ISFETs," *Sensors and Actuators B*, vol. 49, 1998.

[23] J. Grade, A. Barzilai, J. Reynolds, L. Cheng-Hsien, A. Partridge, L. Miller, and J. Podosek, "Low frequency drift in tunnel sensors," in *Solid State Sensors and Actuators, Transducers '97*, 1997.

[24] S. Marco, A. Pardo, F. Davide, C. Di Natale, A. D'Amico, A. Hierlemann, and W. Gopel, "Different strategies for the dynamical calibration of gas sensors," in *Solid State Sensors and Actuators and Eurosensors IX, Transducers '95*, 1995.

[25] A. Chorti and M. Brookes, "A spectral model for RF oscillators with power-law phase noise," *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, vol. 53, no. 9, pp. 1989–1999, Sept. 2006.

[26] A. Chorti, D. Karatzas, N. White, and C. Harris, "Use of the EKF for state dependent drift estimation in weakly nonlinear sensors," *accepted for publication to Sensor Letters*, 2006.

[27] A. Boukabache, P. Pons, G. Blasquez, and Z. Dibi, "Charactersisation and modelling of mismatch of TCRs and their effects on the drift of the offset voltage of piezoresistive pressure sensors," *Sensors and Actuators A Physical*, vol. 84, pp. 292–296, 2000.

[28] S. K. Hong, "Compensation of nonlinear thermal bias drift of resonant rate sensor using fuzzy logic," *Sensors and Actuators A Physical*, vol. 78, pp. 143–148, 1999.

[29] S. Marco, A. Ortega, A. Padro, and J. Samitier, "Gas identification with tin oxide sensor array and self-organizing maps: adaptive correction of sensor drifts," *IEEE Transactions on Instrumentation and Measurement*, vol. 47, no. 1, pp. 316–321, Feb. 1998.

[30] L. T. Vincent and P. P. Khargonekar, "A class of nonlinear filtering problems arising from drifting sensor gains," *IEEE Transactions on Automatic Control*, vol. 44, no. 3, pp. 509–520, Mar. 1999.

[31] S. Brahim-Belhouari, A. Bermak, M. Shi, and P. Chan, "Fast and robust gas identification system using an integrated gas sensor technology and gaussian mixture models," *IEEE Sensors Journal*, vol. 5, no. 6, pp. 1433–1444, Dec. 2005.

[32] A. Perera, N. Papamichail, N. Bârsan, U. Weimar, and S. Marco, "On-line novelty detection by recursive dynamic principal component analysis and gas sensor arrays under drift conditions," *IEEE Sensors Journal*, vol. 6, no. 3, pp. 770–783, June 2006.

[33] M. Kermit and O. Tomic, "Independent component analysis applied on gas sensor array measurement data," *IEEE Sensors Journal*, vol. 3, no. 2, pp. 218–228, Apr. 2003.

[34] H. Moussa and R. Bourquin, "Theory of direct frequency output vibrating gyroscopes," *IEEE Sensors Journal*, vol. 6, no. 2, pp. 310–315, Apr. 2006.

[35] I. Avramov, M. Rapp, S. Kurosawa, P. Krawczak, and E. Radeva, "Gas sensitivity comparison of polymer coated SAW and STW resonators operating at the same acoustic wave length," *IEEE Sensors Journal*, vol. 2, no. 3, pp. 150–159, June 2002.

[36] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, pp. 1066–1076, 1962.

[37] S. Chen, X. Hong, and C. Harris, "Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 34, no. 4, pp. 1708–1717, Aug. 2004.

[38] V. Vapnik, "SVM method of estimating density, conditional probability, and conditional density," in *ISACS 2000*, 2000.

[39] M. Girolami and C. He, "Probability desnity estimation from optimally condensed data samples," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1253–1264, Oct. 2003.

PLACE PHOTO HERE

Dimosthenis Karatzas is a Research Fellow in the School of Electronics and Computer Science, University of Southampton within the Electronics Systems Design (ESD) and Image, Speech and Intelligent Systems (ISIS) research groups.

He received his BSc degree from Aristotle University of Salonica, Greece in 1998 and his PhD degree from the University of Liverpool, UK in 2003. From 2002 to 2004 he worked as a Research Fellow within the Pattern Recognition and Image Analysis group (PRImA) at the University of Liverpool.

Dr Karatzas has worked and published on the fields of Web document analysis, historical document analysis, colour calibration and intelligent sensing. He is a member of the IEEE and the IEEE Computer Society, the SPIE and the British Machine Vision Association.

PLACE PHOTO HERE

Arsenia Chorti received the M. Eng. degree in Electrical Engineering from the University of Patras, Greece, in 1998, the D.E.A. degree in electronics from the University Pierre et Marie Curie - Paris VI, France, in 2000 and the Ph.D. degree in Communications and Signal Processing from Imperial College London, UK, in 2005. She has worked as a Research Fellow in the School of Electronics and Computer Science, University of Southampton, UK, in the area of intelligent sensing.

She is currently a Research Fellow at the Technical University of Crete in the area of remote sensing and Bayesian modeling. Her research interests include multicarrier communication systems - OFDM, stochastic signal processing, density estimation, novelty detection and prediction/estimation algorithms.

PLACE PHOTO HERE

Neil White holds a personal chair in the School of Electronics and Computer Science, University of Southampton He has been active in sensor development since 1985. In 1988 he was awarded a PhD from the University of Southampton. He has considerable experience in the design and fabrication of a wide variety of sensors, formulation of novel thick-film sensing materials and intelligent sensor systems.

Professor White is Director and co-founder of the University of Southampton spin-out-company Perpetuum Ltd. He has over 200 publications in the area of instrumentation and advanced sensor technology. His professional qualifications include Chartered Engineer, Fellow of the IET, Fellow of the IOP, Chartered Physicist and Senior Member of the IEEE.

PLACE PHOTO HERE

Chris J. Harris received the BSc degree from the University of Leicester; MA degree from Oxford University and a PhD and DSc degrees from the University of Southampton.

He has had Professional Appointments at Cranfield and Southampton Universities and Imperial College, as well as being Deputy Chief Scientist in the MOD (UK). Currently he is an Emeritus Professor at the University of Southampton. His current research interests lie in data fusion, data based modeling and nonlinear optimization. He has authored/co-authored 12 research books and over 450 research papers and is the associate editor of numerous international journals.

Dr. Harris was elected to the Royal Academy of Engineering in 1996, awarded the IEE Senior Achievement Medal in 1998 and the 2001 IEE Faraday Medal for distinguished research in control and signal processing.