Guidelines for Managing Sensors in Cyber Physical Systems with Multiple Sensors

Pradeepkumar Ashok, Ganesh Krishnamoorthy, and Delbert Tesar

Robotics Research Group, The University of Texas at Austin, PRC/MERB-Robotics, Building no. 160, Room No. 1.206, 10100 Burnet Road, Austin, TX 78758, USA

Correspondence should be addressed to Pradeepkumar Ashok, pradeepkumar@mail.utexas.edu

Received 23 July 2011; Revised 27 October 2011; Accepted 22 November 2011

Academic Editor: Bernhard Jakoby

Copyright © 2011 Pradeepkumar Ashok et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. Introduction

Cyber physical systems (CPSs) are defined as integrations of computation and physical processes [1] and are characteristics of most large systems. Given the large number of system variables associated with even a moderately complex CPS, a full suite of sensors is often required to enable comprehensive monitoring of the system. There are three main facets to consider in the use of multiple sensors: sensor deployment, sensor assignment, and sensor coordination [2]. These aspects are driven by the combined (and often conflicting) objectives of effectively addressing a system’s feedback needs, keeping the costs and complexity to a minimum without any compromises on the overall reliability or performance of the system. There is thus a need for a criteria-based sensor management framework that can help decide upon and best utilize the available sensing resources at any given time.

In this paper, the objective of sensor management is to enable the selection and coordination of a suite of diverse sensors to monitor a dynamic system using well-defined criteria and a Bayesian network approach for estimating values, combining and propagating uncertainties. With such an approach, it is possible to provide a synergy (via functional redundancy) that can enable the use of different sensors to corroborate each other as well as influence the availability of information in situations where data from specific sensors becomes partially or completely unavailable due to sensor or connector failures, bandwidth/power constraints, and so forth. Criteria-based sensor management can help in the selection and use of a finite set of sensors in concert with available computational resources to maximize the utility of the information available regarding the system at any given time (by providing guidelines that the system designer or operator can use to make decisions such as determining the priority of a sensor or a subset of sensors to address a specific objective). It can also help avoid overwhelming data transfer/computational and memory requirements in a system with multiple sensors by ensuring that only the essential data is acquired and utilized.

2. Literature Review

The term “sensor management” is often used in the context of wireless sensor networks to refer to the process of scheduling and activating the appropriate sensors within a group of sensors distributed over a wide geographical area to address issues like energy consumption, limited bandwidth, and so forth or in the context of target tracking where
it refers to the process of selecting appropriate sensors, modalities, and so forth to optimize their effectiveness in characterizing the probability of a target occurring in a region under consideration [3–6]. The common thread in the examples above is that application specific criteria are used to make decisions on what sensors to use, when, and for which purpose. Many of these applications use criteria/norms derived from the field of information theory (such as entropy, mutual information, and Kullback Leiber or Renyi divergence, etc.) in combination with some form of estimation theory (Kalman filter, particle filters, etc.) [7]. In addition, researchers have also explored alternative approaches to sensor management in other domains such as the use of geometric interaction between sensors and the environment in conjunction with Bayes reasoning for sensor selection in a robotic system [8], the use of a gating neural network and a rules/knowledge database to estimate the reliability of sensor readings and the sensors to be used in a surface grinding process [9], using an empirical Bayes procedure for fault detection in diesel engines [10], using a decision-theoretic approach based on user defined criteria for surveillance [11], and using soft computing/fuzzy logic techniques in aircraft sensor management [12].

Different researchers have explored a diverse range of techniques to approach the problem of fault detection and isolation using analytical redundancy. These include using fuzzy logic [13], the Nadaraya Watson statistical estimator [14], Kalman filtering [15], principal component analysis [16], subspace model identification [17]. Although each method has its own advantages (speed, accuracy, ease of implementation, etc.) and weaknesses (need for a system of mathematical models/equations, inability to detect multiple sensor faults, inability to distinguish between sensor and system faults, need to integrate different approaches together in the same application in order to accomplish different tasks like modeling, fault detection, fault isolation, etc.), the focus of this work is to use the Bayesian causal network framework to accomplish these goals. This approach can provide a unified, data-driven framework for correlating the system variables in a physically meaningful manner (that can also be represented graphically for intuitive understanding) as well as perform fault detection, isolation, and fault accommodation using the same framework. In addition, the existence of a well-developed mathematical formalism based on probability theory helps accounting for the nonlinearities and uncertainties associated with the system under consideration. References [18–22] are examples of such approaches.

The primary objective in creating the network is to combine information from distinct, nonredundant measurements and provide information fault tolerance. Consider a system with multiple sensors such as the electromechanical actuator (EMA) shown in Figure 1, fitted with sensors measuring eleven different phenomena: current, voltage, acoustic noise, torque, angular acceleration, output speed, output position, vibration, temperature, magnetic field, and motor position encoder.

In this system, suppose the torque sensor fails during operation. The torque value can be inferred from the magnetic field sensors provided that an analytical or empirical relationship has been established previously between magnetic field and torque. Taking this one level further, an analytical relationship can possibly be established between all the sensors, an example of which is shown in Figure 2. In this figure, all eleven phenomena measured are linked directly or indirectly to all other sensors. Now even if both the flux density sensor and the torque sensor were faulty, the value of torque could still be inferred from any of the other sensors directly or indirectly linked to the torque sensor.

Figure 2 shows an arbitrarily designed network that satisfies the redundancy requirements and provides fault tolerance. Another possible sensor network is shown in Figure 3. There can be many different ways in which the sensors may be linked to provide for fault tolerance. There is currently no unified set of guidelines to aid in the selection of one network configuration over another.

In this paper, we will address the following two questions that arise for such a system with multiple sensors.

1. In Section 3: what is the best way to associate or link the information obtained from the various sensors?

2. In Section 4: after such an association on network is identified and adopted, what choices exist for the human decision maker on how best to use such a network for maximizing system performance in real time?

3. Network Design Criteria

For most engineering applications, the nodes in the Bayesian network represent the different physical parameters of interest for which sensors are integrated into the system. A network composed of these measurands/variables therefore needs to be designed concurrently with the design of the actual system itself. It needs to mirror the actual system as closely as possible since it is meant to represent the behavior of the system for decision-making during operation. The process thus tends to be iterative, as there are numerous design criteria that need to be balanced simultaneously.

This section discusses some design criteria that may be used not only to determine the choice of sensors while designing the physical system but also to address some of the requirements to create a Bayesian network representation for it as per Pearl’s network construction algorithm [23], that is, determining relevant nodes, their ordering, directing the links appropriately, defining the node parameters, and so forth. (Note that these criteria are only a representative list to provide some guidelines that may be used to create and refine the network. Numerous criteria have been proposed and applied successfully in many applications, for instance, in [24–26] and others. More application-specific criteria may be defined in the future.)

3.1. Relative Importance of Sensors. In any application, there are essential sensors without which it may be impossible to achieve satisfactory system operation, and additionally there may be optional sensors that are used to monitor some secondary parameters of interest to enable enhanced system performance.
In some applications, the sensors corresponding to the critical variables of interest may be too fragile and may be prone to frequent failure or loss of performance (e.g., high-precision position encoders are usually sensitive to high operating temperatures). Any degradation or unexpected loss of information from such a sensor vital to the system may lead to undesirable system behavior or, in the extreme case, a catastrophic system failure.

In such situations, if the sensors are too expensive to replace or are located in an inaccessible location within the system and it is not possible to replace or repair them when the system is in operation without other consequences (altering the system, downtime costs incurred as a result of shutting down the system for repair, etc.), it is desirable to provide some failsafe provision for obtaining these critical measurands, in case of a loss of information from their corresponding sensors.

With the use of a Bayesian network to provide functional redundancy, data from one or more of the other operational sensors can be used to set evidence to the network, and the value of the node corresponding to the sensor of interest, say $S_7$, can be determined using probabilistic inferencing. In terms of the network structure, this means that the node $S_7$ must be related to as many other nodes as possible. The objective is to provide as many alternative sources of information as possible to infer the critical measurands so that failsafe operation is possible. Different network structures can produce data of differing quality. The most suitable network would be one where the value of $S_7$ can be obtained from the node(s) which can be potentially set as evidence, without the need to traverse through a lot of intermediate nodes or links. As an example, consider two possible network structures representing relation between five variables of interest $S_1, S_2, \ldots, S_5$, (Figure 4) with $S_5$
being the most critical measurand. Consider the case where there is a loss of information from the sensor corresponding to $S_5$. From Figure 4(a), it can be seen that the value of the node $S_5$ can be inferred using data from any of the sensors corresponding to nodes $S_1$ through $S_4$ with only one intermediate link involved. The uncertainty in the inferred value of $S_5$ is determined by the relationships $S_5 \rightarrow S_i$ as encoded in the conditional probabilities $P(S_i \mid S_5)$, where $i = 1, 2, 3, 4$. Even if one or more of the other sensors $S_1, \ldots, S_4$ become partially or completely unavailable, an alternative always exists to infer the value of $S_5$ (except in the extreme case where all the sensors $S_1, \ldots, S_4$ become unavailable). However, in Figure 4(b), the best option available to infer the value of $S_5$ with least uncertainty is by setting the value of the sensor corresponding to $S_2$ as evidence to the network. Although any of the other sensors $S_1, \ldots, S_4$ may still be used to infer the value of $S_5$, if the sensor corresponding to $S_2$ also becomes unavailable, the uncertainty in the inferred value will be higher.

**Rule of Thumb 1.** If there is a sensor in the system that is very important for operational reasons, it helps to design the network such that the number of links directly inbound/outbound onto that sensor is maximized.

### 3.2. Causality

The topology of a Bayesian network is highly influenced by the ordering of the nodes that represent the variables in the system under consideration. Strictly speaking, the links in a Bayesian network only represent the conditional independencies between the connected nodes and need not necessarily represent causal relationships between those nodes. However, using causal relations to represent the links between the nodes can help attribute physical meaning to the values that are obtained using the network, making it more intuitive for the user to comprehend those values and use them in decision-making.

For instance, consider a network with two nodes, current and torque, representing a motor. Assume that comprehensive experimental data regarding both the variables is available over the entire operating range in an application where the motor is used and can be used to create the required conditional probability tables or CPTs. The relation between them can be represented as two possible network structures as shown in Figures 5(a) and 5(b). From a mathematical perspective, both of the above networks are equally valid since both forward and inverse probabilistic reasoning based on available information, that is, $P(\text{Torque} \mid \text{Current})$ or $P(\text{Current} \mid \text{Torque})$, are possible by simply using the CPT or Bayes’ theorem, as the case may be. But for both experts (who are involved in designing the system and its Bayesian network representation) and nonexperts (who may be the end users making the final decisions for operating the system), the structure shown in Figure 5(a) will provide a greater intuition in decision-making since it represents what actually happens in a motor, that is, the current applied across the motor windings results in torque generated by the motor (due to the air-gap magnetic field) and not the other way around, with the torque generated being directly proportional to the magnitude of the supplied current.

Various authors including [23, 27, 28] have highlighted the opinion that more often than not, a causal model underlies any real-world joint probability distribution and typically results in a Bayesian network that can be considered practically useful. Pearl [27] emphasizes that use of such causal schema minimizes the number of relationships that need to be considered to model any system, thus resulting in compact networks (i.e., one displaying more independencies than a noncausal representation) with lesser interlinking between the nodes (no unnecessary or redundant links). The simpler structure, in turn, has a significant influence on the inferencing process (more compact CPTs, faster calculations in general). Kenny [29] outlines the following three conditions which may be used to determine whether a variable $A$ causes another variable $B$ or not (and hence also examine the direction of the link between $A$ and $B$ in a Bayesian network).

1. **Precedence in Time.** For a variable $A$ to cause a change in a variable $B$, $A$ must temporally happen before $B$. This implies that the causal relation is asymmetric.
(2) Functional Relationship. There must be a function relationship between the cause and the effect parameters \((B = f(A))\). If the knowledge of one variable does not provide any additional information regarding the other variable, then they can be considered as independent of each other. If not, then they are related.

(3) Nonspuriousness. The relation between \(A\) and \(B\) should not be influenced by the presence of a third variable \(C\) that causes both \(A\) and \(B\) such that if \(C\) is controlled, then \(A\) and \(B\) become independent.

**Rule of Thumb 2.** If human understanding of process is important, then prioritize causal arrangement of sensor network nodes.

3.3. Sensor Reliability. Sensors can be affected by a number of factors in their operational environment. Factors like heat/temperature cycling, mechanical shock/vibrations, humidity, power-on/power-off cycling, and so forth, can sometimes have detrimental effects on the on-board signal processing electronics (e.g., oxidation and failure in solder joints, fretting leading to unreliable contacts, etc. [30, 31]). For sensors not based on a noncontact operating principle, the sensing element may itself undergo wear and tear due to physical contact. In most cases, the data from sensors is sent to a remote data acquisition device or a computer, where it is transformed into useful information (e.g., performance maps) that may be used for decision-making. In this process, data from sensors may become unavailable due to a fault in intermediate connectors or wiring that conveys the sensor output signal to the processor (the analogous situation in case of wireless sensors would be a fault in the transmission link). Most sensors also need a power supply; a fault in the power leads may cause the sensor to become inoperative. All the factors described above may be taken together as representative of how reliable a sensor is.

Fraden [30] defines the reliability of a sensor as its ability to perform its required function under specified conditions for a stated period. Reliability is often expressed as the probability that the sensor will function without failure over a certain time or a specified number of cycles of use. A common metric for specifying reliability indirectly is in terms of mean time between failure (MTBF) which is the average expected time between failures of like units under like conditions (as specified in the MIL-HDBK-217 standard) [31]. It is typically calculated based on installed equipment (MTBF = total time exposure for all installed units/number of failures). Such information is rarely provided in the sensor specifications from manufacturers due to factors like the lack of a standard measure for reliability, the need for accelerated life testing under extreme environmental conditions, and so forth.

However, if such data is available for any system, for example, based on the operational history of the system and the various sensors integrated into it, the knowledge may be used to refine the structure of the Bayesian network for future versions of the system. The nodes corresponding to sensors which are traditionally found to be extremely reliable may be connected to as many other nodes as possible, representing other sensors which may be less reliable, in order to provide a greater assurance of back-up information being available in case of a loss of information from the unreliable sensors. In Figures 6(a) and 6(b), suppose that the sensor corresponding to the node \(S_1\) is considered to be the most reliable amongst all the available sensors. In case one of the sensors \(S_1, S_2, \ldots\), and so forth, becomes unavailable, then the network structures shown can help infer the value of those sensors using the value of \(S_1\) within acceptable limits (depending on the quality of data used to generate the CPTs).

**Rule of Thumb 3.** Design the network such that sensors with higher reliability are attached to as many other sensor nodes as possible.

3.4. Memory Requirement. By exploiting the conditional independencies between the different random variables of interest (embedded explicitly in the network structure in the form of links between the nodes corresponding to the variables), a Bayesian network allows compact storage of their joint probability distribution locally in the form of CPTs for all the nonroot nodes in the network. As seen earlier, if data correlating all the variables is available then there are multiple ways in which the nodes may be connected based on the domain experts’ opinion. However, in doing so, caution must be exercised since different types of connections can result in different CPT configurations for various nodes. The resultant form of the CPTs can have a significant impact on the usefulness of the overall network in addressing the system’s operational goals.

Consider two possible network structures that relate variables of interest in a system \(A, B, C, \) and \(D\). Assume that each variable has two states True or False. In Figure 7(a), the total number of parameters in the CPTs of \(A\) and \(B\) is 2 each whereas the number of parameters in the CPT of \(C\) is
Rule of Thumb 4. Serial networks have lesser memory requirements than parallel networks. So for embedded systems, a serial network structure is best.

3.5. Computational Complexity. With the development of a variety of inferencing algorithms and advances in computational power, the use of Bayesian networks as a tool for both modeling and decision-making has been increasing in many domains for objectives like diagnosis, fault detection, classification, and so forth. The extent to which a system is accurately represented by the model and the quality of results obtained using the model are direct functions of the network structure. For instance, [34] demonstrate that inferencing algorithms are as sensitive to the network structure as the probability values encoded in the different node CPTs [35]. State that the most effective networks are those that combine sound expert knowledge to define the network structure (qualitative) and use extensive data to identify/refine the probability values of the variables represented by the nodes in the network (quantitative). However, despite the value of such a knowledge-based approach [35], there is no prescribed method to construct the network structure when done by domain experts.

The process of creating the network structure based on expert opinion is iterative. A basic structure is first created and then refined based on feedback from other experts (often the direction of links that result from this process imply causality). Then, using the preliminary structure, the network may be implemented under real-world conditions (with components like a graphical user interface, visualization tools, etc., added) to carry out a particular task. This is done to verify its ease of use and intuitiveness in conveying the system characteristics to the end user. Based on user feedback, the network may once again be modified, if necessary, for better usability. If it is found that the results obtained using the network are not satisfactory (or worse, contradictory to those expected based on expert opinion or user experience), its structure may need further refinement. At each iteration, links or nodes may be added to the network or they may be pruned, the direction of some links may be reversed, and so forth. These small changes may or may not always be beneficial. In some cases, they may possibly diminish the efficacy of the network in achieving its intended purpose (since each change may affect factors like the size of node CPTs, type of data/experimentation needed to estimate the CPT parameters, etc.).

Consider a case where a domain expert creates a network for a system with a set X of critical variables and a set Y of variables of secondary importance. In such a case, it would be imperative to represent all the variables in X as nodes in the network, but the expert has to make subjective choices regarding how many/which specific variables from Y also need to be included in the network, if these variables are measurable, their relevance to the variables in X as well as to the goals of creating the network, and so forth. If such a network is intended to be used for real-time operation, then the insertion of numerous additional nodes into the network or a high degree of interlinking between the nodes in X and Y may render it too intractable to satisfy the real-time operation criterion (large CPTs can prove to be a computational hindrance in such cases due to the longer times needed to parse and extract values from the CPTs in inferencing algorithms, especially if the CPT is
sparsely populated, or individual state probabilities are low and widely spread, etc.).

**Rule of Thumb 5.** More nodes in a network imply a greater confidence in the sensors and the system. However they come with a computational overload. The network structure has to be matched to the computation power available.

3.6. **Redundant Sensors.** In some cases, it may be necessary to introduce additional nodes into the network to increase its effectiveness in achieving the application objectives. Consider the network in Figure 8(a) designed for decision-making in a condition monitoring application. Assume that each node in the network represents a sensor corresponding to a domain variable of interest and each link represents a physical process that transforms the variable represented by the parent node to the one represented by the child node. With any unexpected deviations in sensor readings, the challenge facing the decision maker who operates the system is to decide if the variations indicate a potential fault in one or more sensors or whether they are indicative of a fault in the monitored system. If the variations are inadvertently attributed to faulty sensors when in reality, they may be the result of degradation in one of the system’s subcomponents, it can result in a false alarm from the condition-monitoring algorithm that utilizes this network (either calling for unscheduled system maintenance or, in the extreme case, leading to complete catastrophic system failure).

Reference [36] presents a novel Bayesian network-based algorithm to detect and isolate the cause of such deviations (sensor versus system process). The algorithm requires the addition of nodes (representing redundant sensors) to distinguish between sensor and system faults at extremities of the network (root and leaf nodes and the links attached to them). Even though the size of the network increases marginally, the addition of the redundant nodes is critical to achieve the desired functionality in the fault detection and isolation algorithm. Thus, the intended use of the network must always be taken into consideration while designing and before finalizing the structure of the network.

![Figure 8: Use of redundant nodes.](image)

**Rule of Thumb 6.** Redundant sensors modes attached to the network increase confidence in the system.

4. **Operational Criteria**

Once the system design has been completed (with the requisite sensors integrated into the system) and a representative Bayesian network has been designed for it, the next step is to determine suitable criteria that may be used for managing information from all the sensors while the system is in operation. The objective is to make the best use of the information available from the finite set of sensors and the network in conjunction with the available computational resources at any given time. These operational criteria may be used to make decisions regarding how the available sensors may be prioritized to adapt to varying task demands, determine the best options for sensors that may serve as alternatives used to infer the value of failed sensors, determine what sort of information can be gleaned from the network, account for constraints that may arise during operation like limited bandwidth/power, decide on algorithms that are best suited to meet the application constraints, and so forth. Here we assume that the human involvement is present in the decision-making process. This may not be valid for all systems. The following section describes some of these criteria.

4.1. **Node Distance.** Correlating all the variables of interest in the system using a Bayesian network allows the use of any variable to infer the value of any other variable in the network (by setting the former as evidence and using probabilistic propagation to infer the desired value). However, the inferred value (and the uncertainty in it) can be heavily influenced by the number of intermediate links between the evidence node and the query node. Consider the network in Figure 9(a). Suppose the sensor corresponding to node $S_3$ has failed but all the other sensors are operating correctly. Given the network structure, it is possible to use the data from any of the remaining sensors $S_1$ to $S_2$ to set a state of their corresponding nodes as evidence and inferring the value of $S_3$. Intuitively, it can be expected that the uncertainty in the inferred value of $S_3$ will be the least when the value of $S_2$ is used as evidence since there is only one intermediate link between $S_2$ and $S_3$. In this case, the uncertainty in the inferred value is determined by the uncertainty in the process $S_2 \rightarrow S_3$. This relation between $S_2$ and $S_3$ is encoded in the conditional probability distribution of $S_3$, that is, $P(S_3 | S_2)$. Now, if the data from the sensor corresponding to the node $S_1$ is used to infer the value of $S_3$, then the final value is influenced by the uncertainties in two intermediate processes, that is, $S_1 \rightarrow S_2$ and $S_2 \rightarrow S_3$. In this case, the value of the node $S_3$ will be calculated using the chain rule of probability as $P(S_3 | S_1) = P(S_1 | S_2) \cdot P(S_2 | S_3) \cdot P(S_3)$. Since $0 \leq P(\cdot | \cdot) \leq 1$, the value of $P(S_3 | S_1)$ is less than $P(S_3 | S_2)$. In general, in the latter case, the probability distribution is spread over more states of the node $S_3$ with a lower probability value for each individual state. Thus, the farther away the evidence node $S_E$ is from the query node $S_Q$, the greater is the potential uncertainty in the inferred...
Node Distance (ND) may be defined as the shortest possible path between an evidence node \( S_E \) and a query node \( S_Q \) along a directed path between the two.

Using the notation \( \text{ND}_{S_i,S_j} \), the value of node distance can be calculated in terms of the number of intermediate links connecting the sequence of adjacent node pairs between \( S_E \) and \( S_Q \). For instance, in Figure 9(a) considering the nodes \( S_3 \) and \( S_5 \), the node distance is \( \text{ND}_{S_3,S_5} = 2 \). Similarly, \( \text{ND}_{S_1,S_5} = 1 \). As the value of ND increases, the greater is the potential uncertainty in the inferred value. This may be a guideline used by the system operator when determining which of the operational sensors may be used to infer an unavailable value.

However, caution must be exercised since the concept of ND may not work well for certain types of network structures. Consider the network in Figure 9(b). Suppose the sensor corresponding to \( S_3 \) is determined to be faulty. Any of the remaining sensors may be used to determine the value of \( S_3 \). In this case, it must be noted that even though there is only one link connecting any of the nodes \( S_i \), where \( i = 1, 2, 4, 5 \) to \( S_3 \) (i.e., \( \text{ND}_{S_i,S_3} = 1 \)), the uncertainty in the final value of \( S_3 \) will be different depending on which of the nodes is used as evidence. In this case the uncertainty in the inferred value would be dictated by the uncertainty in the relations \( S_3 \rightarrow S_i \) encoded in the respective conditional probability distributions, that is, \( P(S_3 \mid S_i) \). For such network structures, the concept of link strength is more suitable.

**Summary 1.** As an operational criterion, node distance may be used to determine the sensor that is most likely to give a best estimate of another measurand. The smaller the node distance, the better the estimate.

### 4.2. Sensor Health Status

The primary goal of integrating sensors into any system is to provide real-time feedback on the measurands of interest for control purposes and enable the system to successfully accomplish its task (e.g., a quality joint position sensor is crucial for a robot to achieve the desired positioning accuracy in high-precision manufacturing tasks). An equally important task for both the essential and optional sensors in intelligent systems is to enable monitoring of variations in parameters over an extended time by providing reliable and accurate data to periodically update the relevant performance maps. The goal is to track the overall health of the system using condition-based maintenance algorithms [37] to ensure a continued availability of the system as well as to assist the human decision maker in determining the ability of the system to accomplish the required tasks. The implicit assumption for the above objectives is that all sensors are operating as per their design/operational specifications and the data obtained from them is always dependable.

A sensor can be considered “healthy” if it produces an output signal proportional to the input stimulus, within an acceptable amount of deviation as dictated by the sensor physics, resolution, accuracy, application requirements, and so forth. However, as mentioned earlier, the output from the sensors can be affected during regular operation by a number of factors. The effects of these are manifested as undesirable deviations in the sensor output like drift, bias, excessive signal noise, and so forth. Such phenomena may be considered as faults in a sensor that occur intermittently or may occur consistently over an extended period indicating the development of gradual sensor faults. In the extreme case, there may be a complete loss of information from a sensor due to an abrupt failure of the sensing element or its peripherals like power/signal transmission lines, connectors, faults in the onboard signal processing circuits, and so forth. When the required sensor readings become unavailable or when erroneous sensor readings are used for control purposes, it may lead to undesirable system behavior.

Furthermore, using data from faulty sensors to update performance maps, without checking for their validity will result in corruption of the stored maps. This, in turn, may lead to false alarms and missed detection of system faults from the system-level CBM algorithms. In each situation, the health of all the sensors must therefore be taken into account by the system operator in deciding whether or not to utilize the data from a particular sensor. To this end, [36] presents the development of a novel Sensor and Process Fault (SPF) detection and isolation algorithm that can help quantify the trustworthiness of the information from a sensor. Belief values are assigned to the various sensors and processes in the system which is represented using a Bayesian network. Analytical estimates for the various physical quantities represented by the nodes in the network are calculated using standard Bayesian network-inferencing algorithms. By comparing these values against the actual values indicated by the sensors corresponding to those quantities and modifying the belief values based on the results of the comparison, the algorithm provides an indication of the potential source of the fault (i.e., a specific sensor or a group of sensors or a specific process). These belief values provide an intuitive metric representing the health of each sensor that the decision makers can then use in their assessment.

**Summary 2.** Sensor health status is an important criterion that the Human Decision Maker (HDM) could use to disable a failed sensor, so decisions and control are not based on faulty sensor readings. It is very important that sensor failure is distinguished from process degradation and this is enabled by the algorithm presented in [36].
4.3. Resource Availability. In most applications, following some preliminary processing at the sensor-level, the signals from all the sensors monitoring the system are sent to a central location for further processing or for use in deriving higher level information. This configuration is commonly observed in PC-based data acquisition and control of systems like Electromechanical Actuators (EMAs), mobile robots, and so forth. With a limited number of sensors, a point-to-point connection technique is sufficient to connect the sensors directly to the PC without significant design or hardware overhead. However, such an arrangement requires complex cabling arrangements. Hence a bus topology is often utilized wherein all the sensors use a common set of resources for data transmission [38]. In a digital fieldbus system, multiple sensors are connected via shared digital communication lines (thereby reducing the number of cables) to transmit/receive data more efficiently on an as-needed basis [39–41]. When such an arrangement is utilized, the cumulative data bandwidth and latency required for all the sensors being considered play a significant role in the selection of the appropriate bus. This is largely dictated by factors like the type of the sensor output, quantity of output data generated in a specific time period, sampling rate used for the different sensors, mode of acquisition from multiple sensors (simultaneous/multiplexed), and so forth.

Consider, for example, a motor equipped with an incremental encoder producing 10,000 counts per revolution (cpr) and rotating at a moderate speed, say, 600 rpm. This yields an output signal frequency of 0.1 MHz. As the motor speed increases, the volume of output data from the encoder also increases. In addition, the motor may be instrumented with other sensors like current, voltage, temperature, and so forth, which may generate additional volumes of data. To acquire all this information accurately, it needs to be sampled at a high rate. Hence, in addition to the transmission bandwidth, the data acquisition hardware also needs to be capable of handling the frequency requirements for sampling.

With fewer sensors, the total bandwidth requirements are moderate, and it may be possible to sample all the sensors simultaneously with the available data bus and acquisition hardware resources. However, if the system has a large number of sensors which also need to be sampled at high rates, the number of high-speed data acquisition channels required increases (to accommodate the increased bandwidth/sampling requirements) which typically leads to higher overall costs. Often, as a compromise between cost and performance requirements, a limited number of data acquisition channels are used (capable of handling large amounts of data at high frequencies) and the available resources are distributed across all the sensor channels, by using a lower sampling rate, polling the sensors periodically instead of continuous acquisition, and so forth.

The use of a Bayesian network to model the system allows the flexibility of inferring the value of any node/variable in the network (query) using the value of any other node/variable (evidence) in an inferencing process. This capability can be exploited for managing the available resources (bandwidth/sampling rate capability) in certain operating regimes of the system, where it may not be possible to accurately acquire data from sensors with demanding requirements (i.e., those that require a high bandwidth/sampling rate). For instance, in the example cited earlier, if the motor rotates at 6000 rpm, the output frequency from the encoder rises to 1 MHz. If the associated data bus and acquisition hardware are capable of accommodating only 0.5 MHz, it might be more prudent to allocate the available resources to sensors with modest resource requirements, say, the voltage sensors which need to be sampled at only 1 kHz to acquire their output data with the best possible resolution/sampling rates. This data may then be used to infer the values of other variables that have higher bandwidth/sampling rate needs such as motor speed (within reasonable accuracy) using a Bayesian network that includes the motor voltage and speed as nodes.

Summary 3. Different operational regimes utilize different hardware resources. Resource availability is a criterion that can be used in real time to determine the set of sensors that can be enabled or disabled in real time as the situation demands.

4.4. Strength of Relationship between Nodes. The structure of the Bayesian network explicitly represents the conditional dependencies/independencies between the different variables of interest in the system (nodes). The strength of these conditional relationships is encoded in the conditional probability parameters of the CPTs for all the nonroot nodes in the network. However, in any system, a particular set of physical variables, say \(X\), may have a greater influence on a set of variables \(Z\) than another set of variables \(Y\). In such cases, in the scenario that information from one or more sensors corresponding to the variables in \(Z\) becomes unavailable, it would be desirable to use the information available from the sensors corresponding to the variables in \(X\) rather than in the set \(Y\), in order to infer the values of the variables of interest in the set \(Z\).

An approach by which the extent of such influence may be quantified is by using the concept of link and connection strengths. These measures were first introduced by Boerlag [42] for Bayesian networks with binary nodes (two states). The Connection Strength (CS) measures the strength between any two nodes in the network (without accounting for the path between the two) whereas Link Strength (LS) (also referred to as arc weight [43]) specifically calculates the strength along a particular link between two adjacent nodes.

Both these ideas were introduced initially as a way to improve the visualization of the network structure when learned from data (e.g., using thicker links to represent stronger relationships) but were also later used to improve the efficiency of inferencing algorithms (e.g., by eliminating links with insignificant weights) [43, 44]. CS and LS are based on information theory concepts of entropy and mutual information. The entropy and conditional entropy of a discrete random variable are given as follows [44] (upper case denotes a random variables and lower case represents
its states):

\[ U(A) = -\sum_{a_i} P(a_i) \log_2 P(a_i), \]
\[ U(B \mid A) = -\sum_{a_i} P(a_i) U(B \mid a_i). \]  

(1)

The Connection Strength (CS) between any two nodes representing the random variables A and B in the network is defined by how strongly the knowledge of the state of A affects the state of B and vice versa and quantifies it using the concept of mutual information as follows [44]:

\[ CS(A, B) = I(A, B) = U(A) - U(B \mid A) \]
\[ = \sum_{a,b} P(a,b) \log_2 \left( \frac{P(a,b)}{P(a)P(b)} \right). \]  

(2)

The Link Strength (LS) is defined specifically for the relation \( A \rightarrow B \) (i.e., A is the parent and B is its child). If C represents the set of other parents of B, where \( C = \{C_1, C_2, \ldots, C_n\} \) and c represents the set of states of all the nodes \( C_i \), then the link strength is defined [44] as

\[ LS(A \rightarrow B) = \sum_{a,c} P(a,c) \sum_b P(b \mid a,c) \log_2 \left( \frac{P(b \mid a,c)}{P(b \mid c)} \right). \]  

(3)

where \( P_{pr} \) is an approximation of the prior probability of the node being in a particular state and is approximated by averaging the conditional probabilities of that node over all its parent state combinations.

As a computationally useful approximation, [43] provides another formulation of link strength as:

\[ LS(A \rightarrow B) = \sum_{c} P_{pr}(c) \sum_a P_{pr}(a) \sum_b P(b \mid a,c) \log_2 \left( \frac{P(b \mid a,c)}{P(b \mid c)} \right). \]  

(4)

For any application, the values of link strengths and connection strengths may be calculated between different sets of variables and used to determine the most appropriate sensors to use (i.e., if the corresponding nodes have high link/connection strengths indicating that the associated variables are strongly correlated) to infer the information corresponding to faulty or degrading sensors.

**Summary 4.** The connection and link strength criterion may be used to determine the sensor that is most likely to give a best estimate of another measurand. This can be used in conjunction with the node distance criterion to identify the best alternative sensor to derive information when a particular sensor has failed.

4.5. Type of Query. The Bayesian network compactly represents the joint probability distribution of all the variables represented by the nodes in the network. In other words, the network structure and the CPTs for the different nodes represent a comprehensive database that can be queried in different ways to obtain different types of information regarding the system and its sensors. Depending on the application and the operating regime of the system, choosing the right type of query can provide information that is then of greater value to the system operator for decision-making under the given circumstances.

For any Bayesian network, it is possible to define four different types of queries as stated in [33]: probability of evidence, prior and posterior marginal distributions, Maximum A posteriori Hypothesis (MAP), and Most Probable Explanation (MPE). The probability of evidence query refers to the probability of an instantiation \( e \), for a set of evidence variables \( E \) and given as \( P(E = e) \). Such a query may be used to address the duty cycle characteristics of the variables of interest (typically, these are the control variables) in the system. For instance, in Figure 3, if the voltage is known to remain constant at 60 V throughout the operational lifetime of the system, this query may help answer questions of the type “what is the probability of the voltage being 60 V?”

For a joint probability distribution of \( n \) variables, given by \( P(X_1, X_2, \ldots, X_m) \), the marginal distribution is a projection of this distribution on a smaller set of variables (i.e., \( P(x_1, x_2, \ldots, x_m) \), where \( m \leq n \)) and is given by:

\[ P(x_1, x_2, \ldots, x_m) = \sum_{x_{m+1}, \ldots, x_n} P(x_1, x_2, \ldots, x_m). \]  

(5)

In belief updating, the most common type is the posterior marginal query \( P(Q = q \mid E = e) \), where \( Q \) represents one or more query nodes whose values are to be determined, and \( E \) is the set of evidence nodes whose values are observed and instantiated to the values \( e \). This value can be calculated by modifying (6) as follows:

\[ P(x_1, x_2, \ldots, x_m \mid e) = \sum_{x_{m+1}, \ldots, x_n} P(x_1, x_2, \ldots, x_n \mid e). \]  

(6)

Such queries are important when the operator is concerned only about a specific physical variable in the network for decision-making.

For a set of variables \( X \), the MAP query determines the most probable instantiations for a specific set of query variables \( Q \) (where \( Q \subseteq X \)), given the evidence \( E = e \), and is defined as.

\[ MAP(Q, e) = \arg\max_q P(q, e), \]  

(7)

where \( \arg\max_q f(q) \) indicates the assignment of values of \( q \) for which \( f(q) \) is maximal (note that this is not the same as maximizing the posterior marginals of individual nodes). However, it must be noted that there may be more than one such possible set of instantiations which has the maximum probability. MAP queries are important when the operator is concerned about multiple objectives simultaneously and needs to judge if the behavior of a specific group of variables in the network is consistent with other available knowledge for the purpose of decision-making. For instance, in an application such as an EMA used in submarines, the system operator may be concerned with commanding the EMA to generate maximum torque while
producing minimum acoustic noise at the same time. Based on other measurements like current, voltage, and so forth, the operator may be able to decide using the MAP query on a Bayesian network of the system if the projected torque and noise values are plausible.

The MPE query is a special case of the MAP query, where the goal is to determine the most probable instantiations for all the nonevidence variables (i.e., the query variables are all the unobserved, nonevidence variables). In other words, given a set of evidence values $E = e$, the objective is to find a consistent set of values for all the other variables in $X$, that is, $\{X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n\}$ which represent most probable instantiations of those variables. The instantiation $x_1, x_2, \ldots, x_n$ is then referred to as the most probable explanation given the evidence $e$. The MPE query can be defined as

$$\text{MPE}(X, e) = \arg \max_x P(x, e). \tag{8}$$

Again, there may be more than one set of instantiations $x$ that maximize the posterior probability. In general, MPE queries are simpler to determine algorithmically than MAP queries [33]. MPE queries are important when the system operator would like to determine the most probable behavior for all the variables of interest in the system, considered simultaneously, given the knowledge of specific variables (typically the control variables) in the network and examine if they are consistent with the available information for decision-making. This may be important in a task like updating multiple performance maps for CBM of the different components in an EMA based on limited sources of measurable information.

Summary 5. A sensor network gives flexibility to the HDM to pose different queries about the system. Different types of queries have different computational requirements. The HDM should pose queries based on the requirements and the computational constraints.

4.6. Inferencing Algorithm. When a Bayesian network is developed for any application, the resultant topology depends completely on which criteria are accorded importance for that application. For very large or complex systems like aircraft, it may be simpler to design separate networks for each subsystem and analyze them separately, to avoid an overwhelmingly complex network structure for the whole system. When software like AMOS is developed for Bayesian networks-based decision making [45], a diverse suite of inferencing algorithms is therefore typically incorporated into it for utmost flexibility in handling different network topologies and other application-specific operational constraints.

For a simple casual chain (Figure 10(a)), the values for different nodes may be calculated by repeated application of Bayes’ theorem. For polytrees (Figure 10(b)), exact inferencing may be performed by local computations and message passing between nodes (Pearl’s algorithm). For multiply connected networks (Figure 10(c)), exact inferencing may still be applied by using clustering, conditioning, variable elimination, and so forth. Although exact algorithms provide the greatest precision, applying such algorithms to even moderately complex or densely connected networks may require an exorbitant amount of computation/time and memory or in some cases may be unable to even complete the inferencing [46]. In such cases, approximate inferencing is used. Approximate algorithms (stochastic simulations, model simplification, etc.) provide a tradeoff for speed against precision and, for very complex networks, may be the only viable option. Stochastic simulation methods like Likelihood Weighting generate an imprecise answer quickly and refine it iteratively. The accuracy of the results obtained from such algorithms generally improves as the number of samples generated increases.

In some applications, real-time operation is a binding constraint. In such cases, the utility of a computed result is estimated not only on the basis of its accuracy but also on the timeliness with which it is obtained. The utility is considered to degrade (even if the result is highly accurate) as time progresses beyond a predetermined deadline (application-specific) [47]. The extent to which such real-time constraints can be satisfied is, in turn, dictated by factors like choosing the most suitable algorithm for a particular network topology (e.g., both exact and approximate algorithms may be used for the structure in Figure 10(b)), the granularity of discretization for the different nodes (e.g., see [48]), the efficiency of implementation (coding) of different algorithms in the operational software (optimized implementations with efficient coding tend to have faster execution times), and so forth. An important factor in most cases are the computational resources (time/space) available to execute these algorithms.

Though stochastic sampling algorithms are largely impervious to network size or topology, the number of samples that need to be generated to achieve the desired accuracy while satisfying real-time operational constraints may still be a challenge for large networks (the execution time tends to be directly proportional to the number of samples as well as the number of nodes and their interconnections) [46]. For laboratory test setups, it may be possible to implement inferencing algorithms on computers with fast, dedicated processors and ample memory. Alternatively inferencing
code may be implemented via parallel processing on multiple processor cores or distributed/cloud computing. In such cases, it may be feasible to provide real-time operation even for very large networks with highly discretized nodes. However, the same may not be true if the algorithms have to be executed on a system with restricted/shared computational resources (e.g., a mobile robot where the available on-board computation resources have to be shared and allocated among many navigation, control, and communication functions). To account for such scenarios, the decision maker may use one of the many approaches described below to decide on an appropriate compromise between the available computational resources and the operational requirements.

Stochastic sampling algorithms are considered to be anytime algorithms. They can be interrupted at any point in time to yield an approximate result. If these are the algorithms of choice based on the network topology, then the decision maker may vary the number of samples needed (based on suitable convergence criterion like the KL divergence) to satisfy the real-time execution constraint at the expense of some loss of accuracy. Alternatively, the decision maker may also use domain characterization metrics as proposed by [49] as a basis to compare the execution characteristics of different inferencing algorithms on an existing network structure to aid in the selection of the most appropriate one. These include metrics for individual nodes (CPT skewness, maximum and average distance of a node from other nodes, node distance from query nodes) as well as for the entire network (size of state space, number of nodes and links, maximum and average state space size, connectedness [43], maximum and average CPT size, maximum and average number of parents, etc.). The metrics may be used to generate a look-up table beforehand characterizing the performance of the available suite of algorithms and then used by the decision maker to select the most appropriate algorithm based on the task requirements at any given time. A similar approach to prior offline compilation of algorithm execution profiles and their associated mapping to domain characteristics is also proposed [50].

Other techniques may also be used to adaptively allocate the available computing time/memory when the system is in operation. Techniques such as recursive conditioning [51] provide an any-space approach (where the performance of an algorithm improves with increasing memory beyond a minimum requirement) to exact inferencing by utilizing conditioning to decompose a network into smaller subnetworks that are again solved independently and recursively. Reference [52] provides another alternative approach termed as the adaptive conditioning framework to provide a tradeoff between time, space, and quality of results obtained by decomposing the network into subnetworks and allocating different exact/approximate algorithms to process the subnetworks.

Summary 6. The inferencing algorithm should be chosen in real time based on time, space, and accuracy constraints, and the HDM should have access to this choice.

5. Case Study

The guidelines discussed in the previous sections were used to design a sensor network for an experimental actuator test bed at the Robotics Research Group (RRG) laboratory at the University of Texas at Austin. The test bed was designed to be modular, to enable the testing of actuators with different prime movers (Switched Reluctance Motors (SRM), Brushless DC motors, Brushed DC motors, etc.) (Figures 11 and 12).

The National Instruments (NI) Compact Reconfigurable Input Output (cRIO) module sends commutation signals to the H-bridges associated with each phase of the prime mover through an optoisolator circuit.

The primary objective of the test bed is to map the performance capability of an actuator, in particular its torque, efficiency and noise characteristics. The sensors considered for this purpose are voltage, current, torque, speed, and noise sensors. The pulse width modulation (PWM) signal characteristics (i.e., duty cycle and frequency) are monitored directly by the NI-cRIO module. Efficiency is calculated from data obtained from the voltage, current, torque, and speed sensors.

In designing the network, the six main criteria considered were the relative importance of the sensors, causality, sensor reliability, memory requirement, computational complexity, and redundancy. In terms of relative importance, all sensors were considered equal, as this test bed was intended to measure actuator performance. So there was no requirement to intentionally develop a network with a specific sensor as the hub. Causality however was a highly desired criterion (to provide a physical system perspective). So the first network created was based on causality (Figure 13). Here the PWM signal sent from the cRIO board determines the average voltage applied across a motor phase. This in turn dictates the average current flowing through the phase. Current has an effect on both the torque outcome and the noise (due to switching of phases), and this is reflected in the network (Figure 13). The motor torque dictates the speed which also affects the noise.

The next criterion investigated was sensor reliability. Obtaining MTBF (mean time between failures) data from the manufacturers for all the sensors turned out to be very difficult. Since the test bed was designed for use in a controlled environment, the authors did not spend much more time investigating the reliability of each sensor. However, in a system deployed in an uncontrolled environment, if sensor reliability data is not available from the manufacturers, the system designer should preferably try to obtain such data through sensor testing. In this test bed, the data gets continuously streamed from the cRIO to a desktop PC (4 GB RAM, 2.4 GHz). Hence, no modification to the network was needed to accommodate memory requirement or computational complexity. Also, since the test bed was primarily for characterizing actuator performance on a bench setup, fail-safe operation of the sensors was not deemed to be a critical criterion (this may be a critical requirement during the field operation of some actuators). Hence, no redundancy in sensors was considered ([36] provides an example of...
Figure 11: Test bed architecture for testing different electromechanical actuators.

Figure 12: Actuator test bed.

Figure 13: Sensor network.
modifying the network in Figure 13 with the addition of redundant sensors for the explicit purpose of fault detection and isolation). The resultant network therefore remains the same as shown in Figure 13.

6. Conclusion

With the use of multiple sensors to monitor a system like an intelligent EMA, there is a clear need for a criteria-based sensor management framework that can help decide upon and best utilize the available sensing resources at any given time, without compromising on the overall system performance. This encompasses facets like deciding which sensors are needed to respond to changes in the monitored system and its environment, correlating multiple sources of information in the best possible manner, providing redundancy to ensure constant availability of information (under partial or complete sensor failures), and adapting the usage of available sensing and computational resources to changing task requirements while concurrently keeping the costs and complexity involved to a minimum. Founded on the choice of a Bayesian network-based methodology to address the above issues, this paper explored some criteria that could be utilized to improve both the design as well as operational use of such networks.

To enable the domain expert involved in designing the Bayesian network for a system to develop a compact, more usable and efficient network topology, a preliminary list of network design criteria were described in this paper.

Once a representative Bayesian network for a system has been formulated (with the requisite sensors integrated into the system), to assist the system operator make the best use of the available set of sensors and network topology in concert with the available computational resources (prioritizing sensors to adapt to varying task demands, accommodate failed sensors, limited bandwidth availability, inferencing algorithms best suited to meet operational constraints, etc.), a preliminary list of operational criteria was also discussed in this paper. Sensor management is demanding and complex. This paper is a first step towards a criteria-based framework for system data management. Much of the criteria were formulated from first principles and have a theoretic origin. The application of both the design and operational criteria needs to be investigated for a complex system with a multitude of sensors.

Acknowledgment

This work was supported in part by the U.S. Office of Naval Research under Grant no. N00014-06-1-0213.

References

[1] E. A. Lee, “Cyber physical systems: design challenges,” Tech. Rep. UCB/EECS-2008-8, University of California at Berkeley, 2008.

[2] N. Xiong and P. Svensson, “Multi-sensor management for information fusion: issues and approaches,” Information Fusion, vol. 3, no. 2, pp. 163–186, 2002.

[3] T. Onel, C. Ersoy, and H. Deliç, “Information content-based sensor selection and transmission power adjustment for collaborative target tracking,” IEEE Transactions on Mobile Computing, vol. 8, no. 8, Article ID 4745639, pp. 1103–1116, 2009.

[4] M. P. Kolba and L. M. Collins, “Information-theoretic sensor management for multimodal sensing,” in Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS ’06), pp. 3935–3938, August 2006.

[5] D. Vaidya, J. Peng, L. Yang, and J. W. Rozenblit, “A framework for sensor management in wireless and heterogeneous sensor network,” in Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS ’05), pp. 155–162, April 2005.

[6] K. Kreucher, K. Kastella, and A. O. Hero, “A bayesian method for integrated multitarget tracking and sensor management,” in Proceedings of the 6th International Conference of Information Fusion Fusion (FUSION ’08), pp. 1–8, July 2008.

[7] C. Giraud and B. Jouvencel, “Sensor selection: a geometrical approach,” in Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS ’95), vol. 2, pp. 555–560, 1995.

[8] F. Kobayashi, T. Fukui, F. Arai et al., “Sensor selected fusion with sensor selection based gating neural network,” in Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE ’02), pp. 1482–1487, May 2002.

[9] N. Subrahmanya, Y. C. Shin, and P. H. Meckl, “A Bayesian machine learning method for sensor selection and fusion with application to on-board fault diagnostics,” Mechanical Systems and Signal Processing, vol. 24, no. 1, pp. 182–192, 2010.

[10] M. T. Spaan and P. U. Lima, “A decision-theoretic approach to dynamic sensor selection in camera networks,” in Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS ’09), pp. 297–304, September 2009.

[11] M. Oosterom, R. Babuška, and H. B. Verbruggen, “Soft computing applications in aircraft sensor management and flight control law reconfiguration,” IEEE Transactions on Systems, Man and Cybernetics Part C, vol. 32, no. 2, pp. 125–139, 2002.

[12] J. Frolik, M. Abdelrahman, and P. Kandasamy, “A confidence-based approach to the self-validation, fusion and reconstruction of quasi-redundant sensor data,” IEEE Transactions on Instrumentation and Measurement, vol. 56, no. 6, pp. 1761–1769, 2001.

[13] S. J. Wellington, J. K. Atkinson, and R. P. Sion, “Sensor validation and fusion using the nadaraya-watson statistical estimator,” in Proceedings of the 5th International Conference on Information Fusion, vol. 1, pp. 321–326, 2002.

[14] S. Alag, A. M. Agogino, and M. Morjaria, “A methodology for intelligent sensor measurement, validation, fusion, and fault detection for equipment monitoring and diagnostics,” Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 15, no. 4, pp. 307–320, 2001.

[15] R. Dunia and S. J. Qin, “Joint diagnosis of process and sensor faults using principal component analysis,” Control Engineering Practice, vol. 6, no. 4, pp. 457–469, 1998.

[16] L. Jiang, D. Djurdjanovic, and J. Ni, “A new method for sensor degradation detection, isolation and compensation in linear systems,” in Proceedings of the ASME International Mechanical Engineering Congress and Exposition (ASME IMECE 2009), November 2009.
