Applied Soft Computing 42 (2016) 204–214

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Detection and accommodation of outliers in Wireless Sensor Networks within a multi-agent framework

P. Gil [a,b,c,*], H. Martins [b], F. Januário [b,c]

[a] CTS-UNINOVA, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Monte de Caparica, Portugal
[b] NOVA, Departamento de Engenharia Electrotécnica, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Monte de Caparica, Portugal
[c] CISUC – Centre for Informatics and Systems of the University of Coimbra, Universidade de Coimbra, Coimbra, Portugal

ARTICLE INFO

Article history:
Received 8 July 2015
Received in revised form 2 December 2015
Accepted 24 December 2015
Available online 9 February 2016

Keywords:
Outliers detection
Least Squares-Support Vector Machine
PCA
Univariate statistics
Multi-agent systems

ABSTRACT

This paper studies three techniques for outliers detection in the context of Wireless Sensor Networks, including a machine learning technique, a Principal Component Analysis-based methodology and an univariate statistics-based approach. The first methodology is based on a Least Squares-Support Vector Machine technique, together with a sliding window learning. A modification to this approach is also considered in order to improve its performance in non-stationary time-series. The second methodology relies on Principal Component Analysis, along with the robust orthonormal projection approximation subspace tracking with rank-1 modification, while the last approach is based on univariate statistics within an oversampling mechanism. All methods are implemented under a hierarchical multi-agent framework and compared through experiments carried out on a test-bed.

1. Introduction

A Wireless Sensor Network (WSN) is a network comprising tiny, low-cost and low energy sensor nodes, connected to one or more sink devices. Each node is, usually, provided with a wireless radio transceiver, a small micro-controller, a power source and multi-type sensors, such as temperature, humidity or pressure. It can also include analogue-to-digital converter (ADC) and/or digital-to-analogue converter (DAC) ports, as well as a variety of network services, namely localisation, coverage, synchronization, data compression and aggregation, or even security mechanisms [25,27].

This kind of infrastructure is becoming increasingly popular in a number of fields and applications, such as in environmental contexts, habitat or health monitoring, or in military surveillance activities, just to name out a few (see e.g. [22,18]). Because of their inherent constraints, in particular power autonomy, memory, computational power and communication bandwidth, raw data collected from WSNs are quite often unreliable and inaccurate [2,11]. These inaccuracies, generically referred to as outliers in the context of this work, can be regarded as measurements that significantly deviate from the normal pattern of sampled data [28]. For this reason it is recommended that raw data collected through wireless sensor nodes should be purged from outliers.

Outliers detection techniques designed to be implemented on WSNs nodes should have a high detection rate and a low false alarm rate, while presenting a parsimonious consumption of resources. A number of detection methods have been proposed in the last few decades. They can be classified according to the underlying techniques, the network structure or even the type of outliers they can detect (see e.g. [26,25,21]). In what the state-of-the-art recursive methods are concerned, they have not been, to the best of the authors' knowledge, assessed regarding their implementability on sensor nodes and the underlying performance in the context of monitoring systems over WSN, where collected data is quite often non-stationary.

In order to shed some light on this issue, the present work evaluates three different approaches for online detection and accommodation of outliers in raw data over WSNs under a hierarchical multi-agent system based framework. The first approach is a Machine Learning technique relying on a Kernel-based methodology, namely the Least Squares (LS)-Support Vector Machine (SVM), along with an online sliding window scheme [17]. This choice is to some extent motivated by the fact that they do not demand the definition of a probability density function (p_0) for a given hypothesis, they provide computationally efficient decision functions, and they can be applied in high dimensional data sets [7]. To improve the LS-SVM's performance in non-stationary

* Corresponding author at: Departamento de Engenharia Electrotécnica, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Monte de Caparica, Portugal. Tel.: +351 212948545.
E-mail addresses: psg@fct.unl.pt (P. Gil), hmo.martins@campus.fct.unl.pt (H. Martins), fabioj@dei.uc.pt (F. Januário).

http://dx.doi.org/10.1016/j.asoc.2015.12.042
1568-4946/

conditions, the present work considers a modification to the standard method, characterised by redefining the Gaussian kernel. The second methodology relies on Principal Component Analysis (PCA), which usually has a high computational complexity due to the expensive eigendecomposition (ED). To reduce the underlying complexity, this work follows an approach based on a recursive subspace tracking scheme, namely the orthonormal projection approximation subspace tracking (PAST) (OPAST) algorithm [4], as the major subspace is only recursively tracked by using a rank-1 modification. This method is referred to as robust OPAST with rank-1 modification (OPASTr). The last methodology is based on Univariate Statistical Analysis, commonly used in Shewhart control charts. In order to improve its consistency, this technique is implemented under an oversampling framework.

The remainder of this paper is organized as follows. Section 2 presents an introduction to the LS-SVM approach, describes the training algorithm used for online implementation, and presents the proposed modification so as to improve the detection performance in transient time-series. Section 3 provides a brief description with regard to the second technique based on PCA, while Section 4 presents the methodology based on Univariate Statistical Analysis. Section 5 gives a brief overview on the multi-agent framework deployed on the sensor nodes, while Section 6 presents some results and Section 7 concludes this work.

2. Machine learning approach

This section provides a brief introduction to the machine learning technique and describes the proposed modification to improve its performance in transient time-series. The reader is referred to [7] and references therein for a comprehensive description of the standard approach.

2.1. LS-SVND algorithm

The Support Vector Novelty Detection (SVND) method deals with the problem of given a set of vectors $X = \{x_1, \ldots, x_m\} \in \mathcal{X}^m$, such that the sequence x_i, $i = 1, \ldots, m \sim p_0$ (with p_0 unknown) and two hypotheses H_0 and H_1, categorising a new reading $x \in \mathcal{X}$, with identical probability density function p_0, under the underlying two hypotheses. This problem is addressed by defining a decision function $f(x) \in \mathcal{S} \subset \mathcal{X}$ and a real number b, such that $f(x) - b \geq 0$ if $x \in \mathcal{S}$ (x is "normal"), and $f(x) - b < 0$ if x is an outlier. The decision function is designed taking into account the following two constraints:

- Most of the training vectors are assumed to be normal ($X \in \mathcal{S}$), except for a small subset of outliers;
- The bound that surrounds the uncorrupted data should be as small as possible, that is $\mathcal{S} \subset \mathcal{X}$ should have minimum volume.

Based on these constraints, the space of possible functions $f(x)$ is reduced to a Reproducing Kernel Hilbert Space (RKHS) (see e.g. [12,23]), with kernel function $k(\cdot, \cdot)$. This RKHS can be selected by first considering a positive definite kernel function $k(\cdot, \cdot): \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A common choice for the kernel function is the Gaussian Radial Basis Function (RBF) [13], given as:

$$k(x_1, x_2) = \exp\left[-\frac{1}{2\sigma^2}\|x_1 - x_2\|^2\right] \tag{1}$$

where $\|\cdot\|$ represents the canonical norm.

It should be mentioned that a positive definite kernel $k(\cdot, \cdot)$ induces a RKHS, that is a linear space of functions $\mathcal{F}$ represented by a dot product and denoted as $\langle \cdot, \cdot \rangle_{\mathcal{F}}$, with the corresponding norm denoted as $\|\cdot\|_{\mathcal{F}}$. In addition, $\mathcal{F}$ is complete in this norm, and for any $f(\cdot) \in \mathcal{F}$ the reproducing property holds, namely $\langle k(x, \cdot), f(\cdot) \rangle_{\mathcal{F}} = f(\cdot)$.

For a positive definite kernel and the corresponding RKHS $\mathcal{F}$, the SVND method provides the function $f(x)$ as the solution to the following convex optimisation problem, with $0 < \upsilon < 1$ [7]:

$$\begin{aligned} \max_{f(\cdot) \in \mathcal{F}, e_i, b} \quad & -\frac{1}{2}\|f(.)\|^2 - \frac{1}{\upsilon m}\sum_{i=1}^{m} e_i^2 + b \\ \text{subject to} \quad & f(x_i) - b = -e_i, \quad e_i \geq 0 \end{aligned} \tag{2}$$

In (2) the slack variables e_i, along with the constraints, guarantee that the underlying decision function $f_x(\cdot)$ fits the training data, which implies that almost all the training data are located inside the region $\mathcal{S}$. The samples x_i lying outside this region are assumed to be outliers. Further, the number of outliers is kept low by minimizing the term $\sum_{i=1}^{m} e_i^2$, while the term $\|f(.)\|^2$ ensures that the second constraint holds, which results in a minimum volume for $\mathcal{S}$.

The dual minimisation problem associated with (2) is obtained by appealing to a set of Lagrange multipliers $\alpha = \{\alpha_1, \ldots, \alpha_m\}$, with the Lagrangian given as:

$$L = \frac{1}{2}\|f(.)\|^2 + \frac{1}{\upsilon m}\sum_{i=1}^{m} e_i^2 - b - \sum_{i=1}^{m} \alpha_i [f(x_i) - b + e_i] \tag{3}$$

By computing the Lagrangian's partial derivatives with respect to $f(x)$, b, e_i and α_i, and setting them equal to zero, it follows that,

$$\frac{\partial L}{\partial f(.)} = 0 \Rightarrow f(.) = \sum_{i=1}^{m} \alpha_i k(x_i, .) \tag{4}$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{m} \alpha_i = 1 \tag{5}$$

$$\frac{\partial L}{\partial e_i} = 0 \Rightarrow e_i = \frac{\upsilon m}{2}\alpha_i \tag{6}$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \Rightarrow f(x_i) - b + e_i = 0 \tag{7}$$

The above four equations can be rewritten as:

$$\begin{cases} \displaystyle\sum_{j=1}^{m} \alpha_j k(x_j, x_i) - b + \frac{\upsilon m}{2}\alpha_i = 0 \\ \displaystyle\sum_{j=1}^{m} \alpha_j = 1 \end{cases} \tag{8}$$

In a compact form (8) can be described by the following matrix equation:

$$\begin{bmatrix} 0 & I \\ -I^T & H \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{9}$$

where I and α are vectors with length m, while H is a square matrix of size $m \times m$, as follows:

$$I = [1 \cdots 1] \tag{10}$$

$$\alpha = [\alpha_1 \cdots \alpha_m]^T \tag{11}$$

$$H = \begin{bmatrix} k(x_1, x_1) + \dfrac{\upsilon m}{2} & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) + \dfrac{\upsilon m}{2} \end{bmatrix} \tag{12}$$

The optimal decision function $f_x(x)$ is given as the solution of (9), namely

$$f_x(x) = \sum_{i=1}^{m} \alpha_i k(x, x_i) - b \tag{13}$$

with $f_x(x) \geq 0$ when x is a "normal" reading and $f_x(x) < 0$ if x is an outlier.

Since readings collected from a given system are in most cases not clean, i.e. noisy raw data, the above discriminant is rather inefficient in what the sensitivity and specificity of the underlying decision (H_0 or H_1) is concerned. In order to get around this issue, an outlier index I_t was proposed in [7]. At a given time t, the detection algorithm is trained using the m most recent samples, yielding the vector α_t and b_t, and being the outlier index I_t computed according to:

$$I_t = -\log\left[\sum_{i=1}^{m} \alpha_{i,t} k(x_{t-(m+1)+i}, x_t)\right] + \log[b_t] \tag{14}$$

where b_t can be regarded as a scaling factor for α_t, while the subscript $(t-(m+1)+i)$ corresponds to the online sliding window used in the training algorithm. By making use of I_t, a measurement is consider an outlier if $I_t > 0$. In practice, however, in order make I_t less sensitive to noise in raw data it is instead compared to a threshold $\eta > 0$, with $\eta \approx -\log(\eta')$, $\eta' < 1$, $\eta' \approx 1$, typically chosen as 0.99. Interestingly, it can be shown for $m \to \infty$ and $x \in R : x \sim \mathcal{N}(\mu, \varsigma^2)$ and $k(\cdot, \cdot)$ the Gaussian kernel (1) [8] that,

$$I_t \geq \eta \Leftrightarrow \frac{\|x_t - \mu\|}{\varsigma^2} \geq \psi\left(\frac{\sigma}{\varsigma}, \eta, \upsilon\right) \tag{15}$$

with $\psi(\cdot)$ a given threshold. In such conditions (see e.g. [7]), the proposed modified test is equivalent to comparing the distance to the distribution mean to the distribution spread.

2.2. Online algorithm

For online detection of outliers, the training set is updated at each sampling time with a new sample collected from the system, while the oldest one in X is discarded. At time t the training time-series consists of m samples, namely:

$$X = \begin{bmatrix} x_{t-m} & x_{t-m+1} & \cdots & x_{t-1} \end{bmatrix}^{\mathrm{T}} \tag{16}$$

By solving Eq. (9), the following equations hold:

$$b_t = \frac{1}{I \cdot H_t^{-1} \cdot I^{\mathrm{T}}} \tag{17}$$

$$a_t = H_t^{-1} \cdot I^{\mathrm{T}} \cdot b_t \tag{18}$$

In order to compute b_t and a_t the inverse of matrix H_t has to be found.

$$H_t = \begin{bmatrix} f_t & F_t^{\mathrm{T}} \\ F_t & W_t \end{bmatrix} \tag{19}$$

with:

$$f_t = k(x_{t-m}, x_{t-m}) + \frac{\upsilon m}{2} \tag{20}$$

$$F_t = [k(x_{t-m+1}, x_{t-m}) \cdots k(x_{t-1}, x_{t-m})]^{\mathrm{T}} \tag{21}$$

$$W_t = \begin{bmatrix} k(x_{t-m+1}, x_{t-m+1}) + \dfrac{\upsilon m}{2} & \cdots & k(x_{t-m+1}, x_{t-1}) \\ \vdots & \ddots & \vdots \\ k(x_{t-1}, x_{t-m+1}) & \cdots & k(x_{t-1}, x_{t-1}) + \dfrac{\upsilon m}{2} \end{bmatrix} \tag{22}$$

At time $t+1$, H_{t+1} is given by:

$$H_{t+1} = \begin{bmatrix} W_t & V_{t+1} \\ V_{t+1}^{\mathrm{T}} & v_{t+1} \end{bmatrix} \tag{23}$$

with

$$v_{t+1} = k(x_t, x_t) + \frac{\upsilon m}{2} \tag{24}$$

$$V_{t+1} = [k(x_{t-m+1}, x_t) \cdots k(x_{t-1}, x_t)]^{\mathrm{T}} \tag{25}$$

To cope with the complexity of inverting high dimensional block matrices, the matrices H_t^{-1} and H_{t+1}^{-1} are computed by appealing to the Sherman–Woodbury theorem (see e.g. [1]).

Let Z be a symmetrical $n \times n$ matrix described by:

$$Z = \begin{bmatrix} A & u \\ u^{\mathrm{T}} & a \end{bmatrix} = \begin{bmatrix} a & u^{\mathrm{T}} \\ u & A \end{bmatrix} \tag{26}$$

where A is a square matrix and a is a scalar. Then the inverse matrix of Z can be computed as:

$$Z^{-1} = \begin{bmatrix} B & q \\ q^{\mathrm{T}} & \tau \end{bmatrix} \tag{27}$$

with:

$$B = A^{-1} + \tau A^{-1} u u^{\mathrm{T}} A^{-1} \tag{28}$$

$$q = -\tau A^{-1} u \tag{29}$$

$$\tau = \frac{1}{a - u^{\mathrm{T}} A^{-1} u} \tag{30}$$

Taking into account (27), matrices H_t^{-1} and H_{t+1}^{-1} can be calculated as follows:

$$H_t^{-1} = \begin{bmatrix} \tau & h_t \\ h_t^{\mathrm{T}} & G_t \end{bmatrix} \tag{31}$$

with

$$\tau = \frac{1}{f_t - F_t^{\mathrm{T}} W_t^{-1} F_t} \tag{32}$$

$$h_t = -\tau F_t^{\mathrm{T}} W_t^{-1} \tag{33}$$

$$G_t = W_t^{-1} + \tau W_t^{-1} F_t F_t^{\mathrm{T}} W_t^{-1} \tag{34}$$

and

$$H_{t+1}^{-1} = \begin{bmatrix} G_{t+1} & h_{t+1} \\ h_{t+1}^{\mathrm{T}} & \tau \end{bmatrix} \tag{35}$$

where

$$\tau = \frac{1}{v_{t+1} - V_{t+1}^{\mathrm{T}} W_t^{-1} V_{t+1}} \tag{36}$$

$$h_{t+1} = -\tau W_t^{-1} V_{t+1} \tag{37}$$

$$G_{t+1} = W_t^{-1} + \tau W_t^{-1} V_{t+1} V_{t+1}^{\mathrm{T}} W_t^{-1} \tag{38}$$

It follows from (31) and (35) that W_t^{-1} is common to both equations. Finally, taking into account (31),

$$G_t = W_t^{-1} + \frac{1}{\tau} h_t^{\mathrm{T}} h_t \Leftrightarrow W_t^{-1} = G_t - \frac{1}{\tau} h_t{}^{\mathrm{T}} h_t \tag{39}$$

Taking into account (39), the block matrix W_t^{-1} can be calculated from H_t^{-1}, and by replacing in (35) H_{t+1}^{-1} can be recursively updated. The outlier detection technique is outlined in Algorithm 1.

Algorithm 1. Outlier detection.

Input: υ, m
Initialise $X \leftarrow \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix}$
Compute H as in (12)
Calculate H^{-1}
repeat
$x_t \leftarrow$ read_sample
Compute b_t and a_t as in (17) and (18)
Obtain I_t from (14)
if $I_t > \eta$ **then**
x_t is an outlier
end if
Obtain v_{t+1} and V_{t+1} from (24) and (25)
Compute W_t^{-1} as in (39)
Calculate H_{t+1}^{-1} using (35)
Update X by adding x_t and removing the oldest sample
until End_Detection

2.3. Proposed approach

One drawback of the standard approach based on the RBF kernel (1) is associated with the fact that when the system from which the readings are taken is not in a steady state, the outliers detection performance is seriously impacted. This is due to transient response influence on the Euclidean norm $\|x_j - x_{j+1}\|$, which results in an increase of the false positive samples. This fact points out to a modification on the original kernel.

In the proposed formulation, the argument is replaced by the difference to the m-samples trend line. The rationale is propped up on the fact that, by taking the deviation to LS approximation, it makes the underlying discriminant less sensitive to the system's dynamics. The modified kernel takes the following form:

$$k(\tilde{x}_1, \tilde{x}_2) = \exp\left[-\frac{1}{2\sigma^2}\|\tilde{x}_1 - \tilde{x}_2\|^2\right] \tag{40}$$

with $\tilde{x}_t = \|x_t - \hat{x}_t\|$ the deviation from the actual sample x_t to the Least Squares estimate $\hat{x}_t$.

The new kernel leads to a change in the equations used for computing the matrix H, namely (12) and (19)–(25). For new kernel, they are written as:

$$H = \begin{bmatrix} k(\tilde{x}_1, \tilde{x}_1) + \dfrac{\upsilon m}{2} & \cdots & k(\tilde{x}_1, \tilde{x}_m) \\ \vdots & \ddots & \vdots \\ k(\tilde{x}_m, \tilde{x}_1) & \cdots & k(\tilde{x}_m, \tilde{x}_m) + \dfrac{\upsilon m}{2} \end{bmatrix} \tag{41}$$

At time t, H_t is given by:

$$H_t = \begin{bmatrix} f_t & F_t^{\mathrm{T}} \\ F_t & W_t \end{bmatrix} \tag{42}$$

with

$$f_t = k(\tilde{x}_{t-m}, \tilde{x}_{t-m}) + \frac{\upsilon m}{2} \tag{43}$$

$$F_t = [k(\tilde{x}_{t-m+1}, \tilde{x}_{t-m}) \cdots k(\tilde{x}_{t-1}, \tilde{x}_{t-m})]^{\mathrm{T}} \tag{44}$$

$$W_t = \begin{bmatrix} k(\tilde{x}_{t-m+1}, \tilde{x}_{t-m+1}) + \dfrac{\upsilon m}{2} & \cdots & k(\tilde{x}_{t-m+1}, \tilde{x}_{t-1}) \\ \vdots & \ddots & \vdots \\ k(\tilde{x}_{t-1}, \tilde{x}_{t-m+1}) & \cdots & k(\tilde{x}_{t-1}, \tilde{x}_{t-1}) + \dfrac{\upsilon m}{2} \end{bmatrix} \tag{45}$$

while at time $t+1$, H_{t+1} is computed as follows:

$$H_{t+1} = \begin{bmatrix} W_t & V_{t+1} \\ V_{t+1}^{\mathrm{T}} & v_{t+1} \end{bmatrix} \tag{46}$$

with

$$v_{t+1} = k(\tilde{x}_t, \tilde{x}_t) + \frac{\upsilon m}{2} \tag{47}$$

$$V_{t+1} = [k(\tilde{x}_{t-m+1}, \tilde{x}_t) \cdots k(\tilde{x}_{t-1}, \tilde{x}_t)]^{\mathrm{T}} \tag{48}$$

Under this new modified kernel-based approach, tagged outliers are replaced by the corresponding Least Squares prediction. Namely, when $I_t > \eta$ (outlier detected) $x_t = \hat{x}_t$. The overall approach is sketched in Algorithm 2.

Algorithm 2. Proposed outlier detection and accommodation.

Input: υ, m
Initialise $X \leftarrow [\, x_1 \quad \cdots \quad x_m \,]$
Obtain $\hat{X}$ by fitting a curve to X
$\tilde{X} \leftarrow \|X - \hat{X}\|$
Compute H as in (41)
Calculate H^{-1}
repeat
$x_t \leftarrow$ read_sample
Obtain predictor $\hat{x}$ by fitting a curve to X
$\tilde{x}_t \leftarrow \|x_t - \hat{x}_t\|$
Compute b_t and a_t as in (17) and (18)
Obtain I_t from (14)
if $I_t > \eta$ **then**
x_t is an outlier
$x_t \leftarrow \hat{x}_t$ % sample accommodated
end if
Obtain v_{t+1} and V_{t+1} from (47) and (48)
Compute W_t^{-1} as in (39)
Calculate H_{t+1}^{-1} using (35)
Update X by adding x_t and removing the oldest sample
Update $\tilde{X}$ by adding $\tilde{x}_t$ and removing the oldest sample
until End_Detection

3. PCA-based technique

This section provides a brief introduction to the PCA-based approach. The reader is referred to [4] and references therein for a comprehensive description.

3.1. Robust recursive location estimator

In the PCA-based technique, the measurement vector x is usually centred. This implies estimating the mean at each sampling time. A recursive mean estimator of $x(t)$ can take following form:

$$\hat{\mu}(t) = \hat{\mu}(t-1) + \frac{1}{t}(x(t) - \hat{\mu}(t-1)) \tag{49}$$

with $\hat{\mu}(t)$ the recursive mean and $x(t)$ the raw measurement vector at time t. In the case of online implementation, a forgetting factor β, with $\beta < 1$ and $\beta \approx 1$, is included in order to give less weight to older samples, so as to cope with changes in the system's dynamics. The resulting estimator is given by:

$$\hat{\mu}(t) = \beta\hat{\mu}(t-1) + (1-\beta)x(t) \tag{50}$$

Given the estimator's sensitivity to outliers in the data, a better choice is to consider the median with regard to a L-samples window:

$$\hat{\mu}(t) = \beta\hat{\mu}(t-1) + (1-\beta)med(A(x(t))) \tag{51}$$

where $A(x(t)) = \{x(t-L+1), \cdots, x(t)\}$, with L the length of the window, and $med(\cdot)$ the median operator. The length L of the estimation

window is commonly kept small to reduce the underlying computational burden. For larger windows, the pseudomedian should be considered instead (see e.g. [20]). Finally, the measurement vector is centred by means of (52).

$$\bar{x}(t) = x(t) - \hat{\mu}(t) \tag{52}$$

3.2. Subspace tracking approach

Principal Component Analysis-based algorithms usually require the computation of the entire ED, which turns out to be computationally demanding, and it is not recommended for online implementation, particularly on sensor nodes. As such, this work takes advantage of subspace tracking, which recursively tracks the signal subspace spanned by the major Principal Component (PC)s $U_B(t)$, instead of computing the whole ED. The PAST algorithm (see e.g. [6]) estimates the signal subspace recursively by minimizing the following cost function:

$$J(W(t)) = \sum_{i=1}^{t} \beta^{t-1} \|\bar{x}(i) - W(t)y(i)\|_2^2 \tag{53}$$

where $y(i) = W^T(t)\bar{x}(i)$ and $J(W(t))$ represents the energy in $\bar{x}(i)$ that is outside the subspace $W(t)$. The subspace $W(t)$ is equal to the major PCs $U_B(t)$ up to an orthogonal transformation or rotation and therefore $W(t)W^T(t) = U_B(t)U_B^T(t)$.

In the PAST algorithm the approximation $\bar{y}(i) \approx W^T(i-1)\bar{x}(i)$ is chosen so that the objective function (53) is relaxed to a quadratic function in $W(t)$, for which a standard recursive LS algorithm can be used to obtain $W(t)$, with low complexity. In order to guarantee the orthonormality of $W(t)$, an additional orthonormalisation is added to the PAST algorithm, being this formulation denoted as OPAST.

To apply the OPAST algorithm in the context of outliers detection, an initial ED has to be available, either taken from an initial data block or by offline estimation. The eigenvalues from the initial ED are then used to estimate the dimension B of the signal subspace, by appealing to a given stopping rule [15], such as the Kaiser's rule [16]. The OPAST algorithm, is outlined in Algorithm 3.

Algorithm 3. OPAST algorithm.

Input: Initial data block $X(0)$
Initialisation:
Initialise β % forgetting factor
Initialise $\hat{\mu}(0) \leftarrow mean(X(0))$
Estimate subspace dimension B with Kaiser's rule
$N_0 \leftarrow length(X(0))$
$C_{xx}(0) \leftarrow \frac{1}{N_0} X^T(0)X(0)$
Obtain $U_B(0)$ and $\Lambda_B(0)$ from the ED of C_{xx}
$W(0) \leftarrow U_B(0)$
$C_{yy}(0) \leftarrow W^T(0)C_{xx}(0)W(0)$
$\Omega(0) \leftarrow C_{yy}^{-1}(0)$
Recursion:
repeat
$y(t) \leftarrow W^T(t-1)\bar{x}(t)$
$g(t) \leftarrow \frac{1}{\beta}\Omega(t-1)y(t)$
$\gamma(t) \leftarrow \frac{1}{1+y^H(t)g(t)}$
$p(t) \leftarrow \gamma(t)(\bar{x}(t) - W(t-1)y(t))$
Orthonormalisation Step:
$\tau(t) \leftarrow \frac{1}{\|g(t)\|_2^2}\left(\frac{1}{\sqrt{1+\|p(t)\|_2^2\|g(t)\|_2^2}} - 1\right)$
$p' \leftarrow \tau(t)W(t-1)g(t) + (1+\tau(t)\|g(t)\|_2^2)p(t)$
Update:
$W(t) \leftarrow W(t-1) + p'(t)g^T(t)$
$\Omega(t) \leftarrow \frac{1}{\beta}\Omega(t-1) - \gamma(t)g(t)g^T(t)$
until End_Time_Series

3.3. Recursive ED computation

Recall that in the PAST algorithm the estimated subspace $W(t)$ is equal to the major PCs $U_B(t)$, up to an orthogonal transformation or rotation $Q(t)$, that is

$$U_B(t) = W(t)Q(t) \tag{54}$$

with $Q(t)$ an orthogonal matrix of size $B \times B$ satisfying $Q(t)Q^T(t) = I$.

To recursively estimate the eigenstructure associated with the underlying subspace, this work relies on the rank-1 modification.

Let the centred measurement $\bar{x}(t)$ projection onto $W(t)$ be given as:

$$y(t) = W^T(t)\bar{x}(t) \tag{55}$$

with the corresponding correlation matrix given by:

$$C_{yy}(t) = E[y(t)y^T(t)] \tag{56}$$

If $W(t)$ is slow time-varying, then the correlation matrix can take the following form:

$$C_{yy}(t) = W^T(t)C_{xx}(t)W(t) \tag{57}$$

Then, by projecting $C_{xx}(t)$ onto the signal subspace $W(t)$, and taking into account (54), the correlation matrix can be written as:

$$C_{yy}(t) = Q(t)\Lambda_B(t)Q^T(t) \tag{58}$$

Through this projection only the major eigenvectors, which span the signal subspace are retained. Therefore, the eigenvectors of the transformation $Q(t)$ can be obtained using the ED of $C_{yy}(t)$. Nevertheless, this method is still rather complex for being implemented on sensor nodes. As such, the ED of the rank-1 update is recursively computed.

In the OPASTr algorithm, the correlation matrix C_{yy} in (58) can be recursively updated according to

$$C_{yy}(t) = \beta C_{yy}(t-1) + (1-\beta)y(t)y^T(t) \tag{59}$$

with β a forgetting factor and $C_{yy}(t-1) = Q(t-1)\Lambda_B(t-1)Q^T(t-1)$, as in (58). Eq. (59) can be rewritten as a rank-1 modification as follows,

$$C_{yy}(t) = Q(t-1)[\beta\Lambda_B(t-1) + (1-\beta)z(t)z^T(t)]Q^T(t-1) \tag{60}$$

with $z(t) = Q^T(t-1)y(t)$.

Consider the corresponding ED described by:

$$\beta\Lambda_B(t-1) + (1-\beta)z(t)z^T(t) = \tilde{Q}(t)\Lambda_B(t)\tilde{Q}^T(t) \tag{61}$$

The term inside the square brackets of (60) is the rank-1 modification, and the eigenvectors of the correlation matrix C_{yy} can retrieved by means of the following relationship:

$$Q(t) = Q(t-1)\tilde{Q}^T(t) \tag{62}$$

Finally, taking into account (62), the major PCs U_B can be obtained through (54).

3.4. Robust recursive detection criteria

The Squared Prediction Error (SPE) and the T^2 score are two measures commonly used in comparing the distance between $x(t)$ and the remaining time-series. For a centred measurement vector $\bar{x}(t)$, the corresponding SPE is given as,

$$SPE(t) = \|\bar{x}(t) - U_B(t)U_B^T(t)\bar{x}(t)\|_2^2 \tag{63}$$

where $U_B(t)U_B^T(t)$ is replaced with $W(t)W^T(t)$. Regarding the T^2 score, its computation can be carried out through the following expression:

$$T^2(t) = \|\bar{x}^T(t)U_B(t)\Lambda_B^{-1}(t)U_B^T(t)\bar{x}(t)\|_2^2 \tag{64}$$

These measures are then compared to predefined thresholds, namely Γ_{SPE} and Γ_{T^2}, in order to detect possible outlying samples. This method is dependent on the selection of an appropriate threshold, which should be recursively updated to allow its adaptation to changes in the system dynamics. Since outliers can impact the threshold update, a robust version of SPE and T^2 score will be considered. The detection of outliers is then based on the following discriminants:

$$|SPE(t) - \hat{\mu}_{SPE}(t)| \geq \xi \hat{\sigma}_{SPE}(t) \tag{65}$$

$$|T^2(t) - \hat{\mu}_{T^2}(t)| \geq \xi \hat{\sigma}_{T^2}(t) \tag{66}$$

where $\hat{\mu}_{SPE}$ and $\hat{\mu}_{T^2}$ are, respectively, robust location estimators of $SPE(t)$ and $T^2(t)$, $\hat{\sigma}_{SPE}$ and $\hat{\sigma}_{T^2}$ are robust scale estimators of $SPE(t)$ and $T^2(t)$ and ξ is the threshold quartile parameter [4]. These robust estimates are computed according to:

$$\hat{\mu}_{SPE}(t) = \beta_{\hat{\mu}_{SPE}} \hat{\mu}_{SPE}(t-1) + (1 - \beta_{\hat{\mu}_{SPE}}) med(A(SPE(t))) \tag{67}$$

$$\hat{\sigma}^2_{SPE}(t) = \beta_{\hat{\sigma}^2_{SPE}} \hat{\sigma}^2_{SPE}(t-1) + c(1 - \beta_{\hat{\sigma}^2_{SPE}}) med(A(\Delta^2_{SPE}(t))) \tag{68}$$

$$\hat{\mu}_{T^2}(t) = \beta_{\hat{\mu}_{T^2}} \hat{\mu}_{T^2}(t-1) + (1 - \beta_{\hat{\mu}_{T^2}}) med(A(T^2(t))) \tag{69}$$

$$\hat{\sigma}^2_{T^2}(t) = \beta_{\hat{\sigma}^2_{T^2}} \hat{\sigma}^2_{T^2}(t-1) + c(1 - \beta_{\hat{\sigma}^2_{T^2}}) med(A(\Delta^2_{T^2}(t))) \tag{70}$$

with $\beta_{\hat{\mu}_{SPE}}$, $\beta_{\hat{\sigma}^2_{SPE}}$, $\beta_{\hat{\mu}_{T^2}}$ and $\beta_{\hat{\sigma}^2_{T^2}}$ forgetting factors close to but smaller than one, $c = 2.13$ is a correction factor for a Gaussian input, $med(\cdot)$ is the median operator, $A(\cdot)$ as defined in Section 3.1 with argument, respectively, $SPE(t)$, $\Delta^2_{SPE}(t)$, $T^2(t)$ and $\Delta^2_{T^2}(t)$. Finally, $\Delta^2_{SPE}(t)$ and $\Delta^2_{T^2}(t)$ are innovations with regard to the underlying estimates, which are given as:

$$\Delta_{SPE}(t) = SPE(t) - \hat{\mu}_{SPE}(t) \tag{71}$$

$$\Delta_{T^2}(t) = T^2(t) - \hat{\mu}_{T^2}(t) \tag{72}$$

3.5. Robust subspace tracking

The previous approach is still sensitive to impulsive outliers. For this reason the robust OPASTr (R-OPASTr) makes use of a robust weight function $\psi(t)$, which affects the updating procedure of the signal subspace, namely when an outlying sample is detected, the updating step is skipped. In this work $\psi(t)$ is chosen as

$$\psi(t) = q_{SPE}(t) \cdot q_{T^2}(t) \tag{73}$$

with

$$q_{SPE}(t) = \begin{cases} 1 & \text{if} |\Delta_{SPE}(t)| < \Gamma_{SPE} \\ 0 & \text{otherwise} \end{cases} \tag{74}$$

$$q_{T^2}(t) = \begin{cases} 1 & \text{if} |\Delta_{T^2}(t)| < \Gamma_{T^2} \\ 0 & \text{otherwise} \end{cases} \tag{75}$$

where $\Delta_{SPE}(t)$ and $\Delta_{T^2}(t)$ are, respectively, given by (71) and (72). Γ_{SPE} and Γ_{T^2} are thresholds recursively updated according to:

$$\Gamma_{SPE} = \xi \hat{\sigma}_{SPE}(t) \tag{76}$$

$$\Gamma_{T^2} = \xi \hat{\sigma}_{T^2}(t) \tag{77}$$

Taking into account $\psi(t)$, the cost function (53) can be rewritten as follows:

$$J(W(t)) = \sum_{i=1}^{t} \beta^{t-1} \psi(i) \|\overline{x}(i) - W(t)\overline{y}(i)\|_2^2 \tag{78}$$

where $\overline{y}(i) \approx W^T(i-1)\overline{x}(t)$ is the projection approximation defined in Section 3.2. The R-OPASTr approach is sketched in Algorithm 4.

Algorithm 4. R-OPASTr algorithm.

Input: Initial data block $X(0)$
Initialisation:
Initialise $\mu(0)$, $U_B(0)$, $\Lambda_B(0)$, $W(0)$, $C_{yy}(0)$, $\Omega(0)$ and β as in Algorithm 3
Initialise $\beta_{\mu_{SPE}}$, $\beta_{\sigma^2_{SPE}}$, $\beta_{\mu_{T^2}}$ and $\beta_{\sigma^2_{T^2}}$
Initialise $Q(0)$ as the eigenvectors of the ED of $C_{yy}(0)$
Recursion:
repeat
$\hat{\mu}(t) \leftarrow$ Eq. (51)
$\overline{x}(t) \leftarrow$ Eq. (52)
Update $W(t)$ and $\Omega(t)$ using Algorithm 3 % OPAST
Robust *SPE*:
$SPE(t) \leftarrow$ Eq. (63)
$\hat{\mu}_{SPE}(t) \leftarrow$ Eq. (67)
$\Delta_{SPE}(t) \leftarrow$ Eq. (71)
$\hat{\sigma}^2_{SPE}(t) \leftarrow$ Eq. (68)
Robust T^2 score:
$z(t) \leftarrow Q^T(t-1)y(t)$
Compute $\tilde{Q}(t)$ and $\Lambda_B(t)$ from the ED of (61)
$Q(t) \leftarrow$ Eq. (62)
$U_B(t) \leftarrow$ Eq. (54)
$T^2(t) \leftarrow$ Eq. (64)
$\hat{\mu}_{T^2}(t) \leftarrow$ Eq. (69)
$\Delta_{T^2}(t) \leftarrow$ Eq. (72)
$\hat{\sigma}^2_{T^2}(t) \leftarrow$ Eq. (70)
Robust subspace update:
$\Gamma_{SPE} \leftarrow$ Eq. (76)
$q_{SPE}(t) \leftarrow$ Eq. (74)
$\Gamma_{T^2} \leftarrow$ Eq. (77)
$q_{T^2}(t) \leftarrow$ Eq. (75)
$\psi(t) \leftarrow$ Eq. (73)
if $\psi(t) = 0$ **then**
$W(t) \leftarrow W(t-1)$
$\Omega(t) \leftarrow \Omega(t-1)$
$\Lambda_B(t) \leftarrow \Lambda_B(t-1)$
$U_B(t) \leftarrow U_B(t-1)$
end if
until End_Time_Series

4. Univariate statistical outlier detection and accommodation technique

This section introduces succinctly the univariate statistics-based method. The reader is referred to [10] and references therein for a comprehensive description regarding this technique.

4.1. Univariate statistical approach

The univariate statistical-based approach to limit sensing assumes that a single variable is observed through sensor readings, and used to determine boundary thresholds for in-control operation [5]. The violation of these limits by observed data is indicative of possible outliers. This methodology is typically employed in Shewhart control charts, and commonly referred to as limit sensing or limit value checking. The upper (Δ_u) and lower (Δ_l) bounds on the Shewhart chart are critical to minimize the rates of false outliers and missed detection.

Statistical hypothesis theory can be used to predict false outlier and missed detection rates on a given ensemble, given a set of thresholds. Consider for a monitoring variable z that any deviations from its mean, $\overline{z}$, are due to additive errors, and the variability of z follows a Gaussian distribution $\mathcal{N}(\overline{z}, \sigma^2)$, with standard deviation σ. Then the probability P that z is within a given interval is given by:

$$P\{z < (\overline{z} - c_{\alpha/2}\sigma)\} = P\{z > (\overline{z} + c_{\alpha/2}\sigma)\} = \frac{\alpha}{2} \tag{79}$$

$$P\{(\overline{z} - c_{\alpha/2}\sigma) \leq z \leq (\overline{z} + c_{\alpha/2}\sigma)\} = 1 - \alpha \tag{80}$$

where $c_{\alpha/2}$ is the standard normal deviation corresponding to the $(1-\alpha/2)$ percentile, and α the level of significance, which specifies the degree of trade-off between false outliers and missed detection rate. Typical values for the standard normal deviation include $c_{\alpha/2} = 1.0\,;\,1.5\,;\,3.0$.

A drawback of monitoring schemes based on the Shewhart control chart is that they are quite sensitive to the thresholds, which ultimately compromises the detection performance. Several methods based on the incorporation of samples from multiple consecutive instantiations can improve the false outlier and missed detection rates, namely the cumulative sum control chart [14] and the exponentially weighted moving average control chart [19]. However, these methods are better suited for small persistent mean shifts.

This work assumes that the deterministic-stochastic process is stationary and the ergodicity is held within each sampling interval $[kT_s, (k+1)T_s]$, with T_s the sampling period and k the discrete time. Additionally, instead of relying on samples taken at each sampling time $k \cdot T_s$, the methodology makes use of an oversampling mechanism ($T_{os} \ll T_s$) within the sampling interval, and including $(k+1)T_s$ in these readings. By means of this scheme, a statistically consistent dataset is collected, which is subsequently used within the univariate statistical approach based on the Shewhart control chart. If the sample taken at time $(k+1)T_s$ is outside the threshold limits, then the sample is replaced by the mean of the oversampling data set.

4.2. Extension to transient time-series

The standard approach discussed in the previous section assumes stationary conditions. When steady-state conditions do not hold, the observations can be assumed as taken from a non-stationary random process with deterministic transient behaviour, corrupted with an ergodic random variable (additive noise).

If the sampling period T_s is adequately chosen, taking into account the bandwidth of the system, the deterministic-stochastic time series for $\tau \in [kT_s, (k+1)T_s]$ can be approximated by means of a linear regression, as follows:

$$z(\tau) = a + b \cdot \tau \tag{81}$$

The computation of the deterministic parameters in (81) is carried out by minimizing the χ^2 merit function, given as:

$$\chi^2(a,b) = \sum_{i=1}^{N_s} \left(\frac{z(\tau_i) - a - b \cdot \tau_i}{\sigma_i} \right)^2 \tag{82}$$

with N_s the number of samples and σ_i the standard deviation.

Since the uncertainty associated with each measurement included in the dataset is not *a priori* known, some considerations concerning the χ^2 fitting have to be taken, in order to derive a plausible value for σ_i. If it is assumed that all samples have equal standard deviation ($\sigma_i = \sigma$) and the model fits sufficiently well, then it is possible to assign an arbitrary value to the standard deviation σ, namely $\sigma = 1$. In this case, the χ^2 merit function can be rewritten as the following residual sum of squares:

$$\chi^2(a,b) = \sum_{i=1}^{N_s} (z(\tau_i) - a - b.\tau_i)^2 \tag{83}$$

Considering the operator $S(\cdot) \triangleq \sum_{i=1}^{N_s} (\cdot)$ and

$$\begin{aligned} S(t) &= \sum_{j=1}^{N_s} t_j \quad S(z) = \sum_{j=1}^{N_s} z_j \\ S(t^2) &= \sum_{j=1}^{N_s} (t_j)^2 \quad S(t \cdot z) = \sum_{j=1}^{N_s} t_j \cdot z_j \end{aligned} \tag{84}$$

the computation of $\theta = [a\ b]^{\mathrm{T}}$ can be carried out according to

$$\theta = \frac{\begin{pmatrix} S(t^2) & -S(t) \\ -S(t) & 1 \end{pmatrix}}{S(t^2) + S^2(t)} \begin{pmatrix} S(z) \\ S(t \cdot z) \end{pmatrix} \tag{85}$$

If a sensor reading taken at $t = (k+1)T_s$ is outside the threshold limits, then it is tagged as an outlier, and accordingly replaced with the trend of the oversampling data, which is computed taking into account (81), for $\tau = (k+1)T_s$. This methodology is outlined in Algorithm 5.

Algorithm 5. Outlier detection and accommodation in transient time series.

Input: f_s, f_{os}, snd
// f_s – Sampling Frequency
// f_{os} – Oversampling Frequency
// snd – Standard Normal Deviate
Output Sample/Accommodated Sample
$T_s \leftarrow 1/f_s$ // sampling period
$T_{os} \leftarrow 1/f_{os}$ // oversampling period
$t_i \leftarrow GetTime$ // initial time
$i \leftarrow 0$ // initial iteration
repeat // oversampling until next sample
  $i \leftarrow i+1$ // iteration
  $z[i] \leftarrow GetSample$ // oversample
  $Sleep(T_{os})$ // node sleep for T_{os} seconds
until $GetTime \geq t_i + T_s$
$y \leftarrow GetSample$ // sample
$(a, b) \leftarrow DataFitting(z)$ // a and b from Eq. (85)
$z[i+1] \leftarrow a + b \times (i+1)$ // next value from linear regression
$\Delta_u \leftarrow z[i+1] + snd \times STD(z)$ // upper threshold
$\Delta_l \leftarrow z[i+1] - snd \times STD(z)$ // lower threshold
if $y \in [\Delta_l, \Delta_u]$ **then** // the sample is not an outlier
  return y // return sample
else // the sample is an outlier
  return $z[i+1]$ // return the accommodated sample
end if

5. Overall architecture

The detection architecture is composed of four main components (see Fig. 1), including a multi input and multi output (MIMO) plant under monitoring, a computer running the required middleware, along with a dispatcher, which feeds the data stemming from the communication infrastructure, based on a IPv6 WSN, to top-level applications, namely the monitoring framework.

Each node collects information from the environment, through attached sensors or transmitters, and subsequently sends it along with generated reports to a router or sink. In addition, each sensor

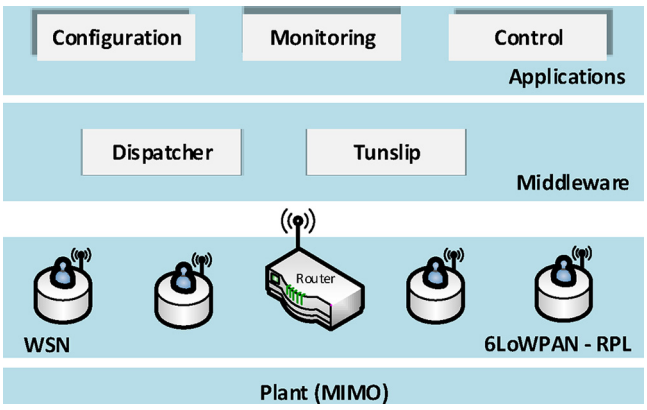

Fig. 1. System architecture.

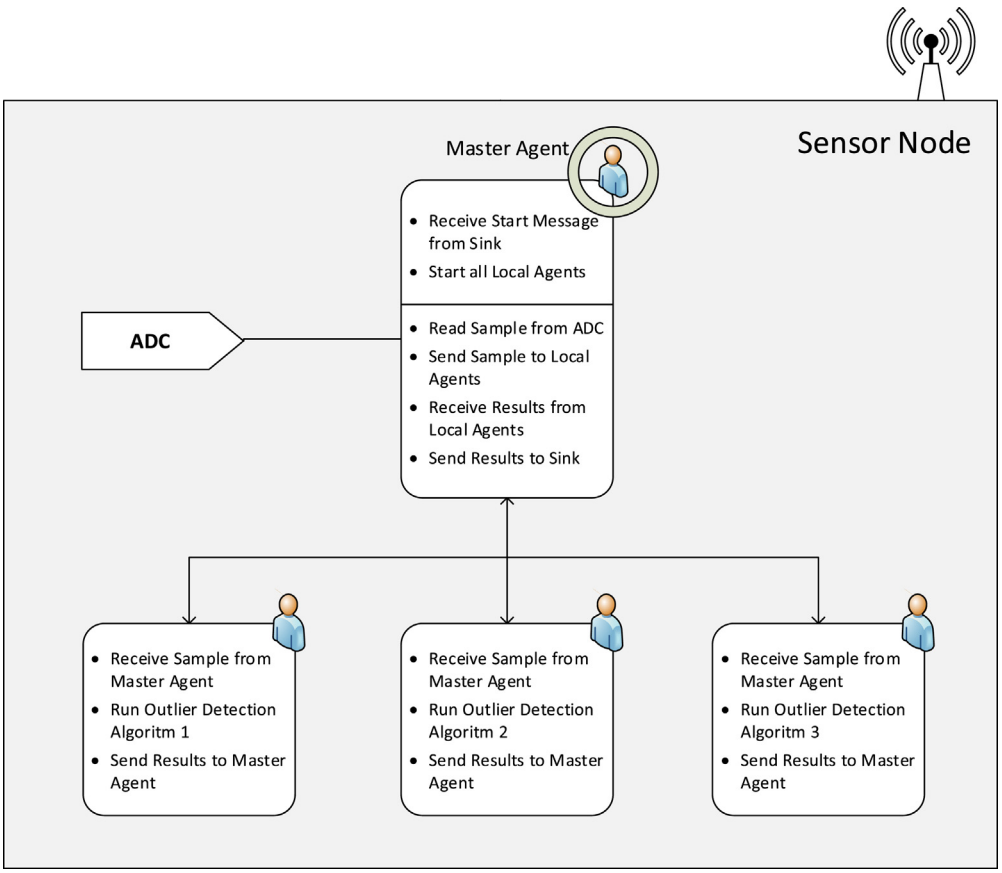

Fig. 2. Illustration of a multi-agent system.

Table 1
Message payload.

Message Type	Node ID	Control ID	Data ID	Agent ID	Agent MSG

node is provided with a set of dedicated detection and accommodation functionalities, implemented by means of mobile agents.

The communications between the router/sink and the applications are carried out in the middleware layer. The Tunslip (see e.g. [3]) creates aSerial Line Protocol (SLIP) tunnel between the "physical" serial port and the "virtual" network interface. The dispatcher software translates IPv6 packets into IPv4 format in order to allow the communication with IPv4 based services and servers. Each message comprises a header and a payload. The message payload (see Table 1) consists of *Message Type*: the message can be originated from the system's application or from a local agent; *Node ID*: denoting the node address; *Control ID*: the command flag for local agents; *Data ID*: data collected in the node ID; *Agent ID*: agent's identifier to be launched, stopped or resumed; *Agent MSG*: data provided by an agent.

The hierarchical multi-agent based architecture is composed of two layers, namely a higher-level layer with coordination functionalities and a bottom layer comprising subordinate agents. These sort of agents are committed to specific tasks, such as monitoring ADCs readings, or detecting and accommodating outliers (see Fig. 2).

Concerning the agents' commands, the platform is provided with: (i) *Start Agent*: starts a monitoring agent by sending the command flag, the agent's ID and the node destination address to the underlying sensor node; (ii) *Stop Agent*: stops a particular monitoring agent; (iii) *Start All Agents*: starts all monitoring agents stacked at the sensor node memory; (iv) *Stop All Agents*: stops all monitoring agents running on a given sensor node.

The main goal of the master agent is to carry out management routines related to subordinate local agents and to coordinate the communications between the node and the sink. When this agent is activated, it automatically launches dependent lower-level agents. This agent is also responsible for monitoring the status of all local dependent agents and, in case of an agent's crash, the master loads to the node's memory a copy of underlying agent.

Monitoring agents are responsible for collecting data from the environment and for accommodating possible outliers in raw data. Each monitoring agent implements one of the approaches considered in this work. If a given sample taken at a given discrete time is tagged as an outlier, the corresponding value is replaced by the accommodated sample and an alarm sent to the sink.

6. Case study

In this section the approaches under assessment are compared in terms detection sensitivity and specificity using a test-bed.

6.1. Test-bed description

The test-bed consists of a benchmark three-tank system, two remote desktop computers, one devoted to the high-level applications (PC2), such as the process monitoring, and the other for running the middleware, namely the Tunslip and the dispatcher (PC1). In addition four wireless sensor nodes are used for building a WSN.

The AMIRA® DTS 200 benchmark three-tank system (Fig. 3) consists of three Plexiglas cylindrical tanks, with identical

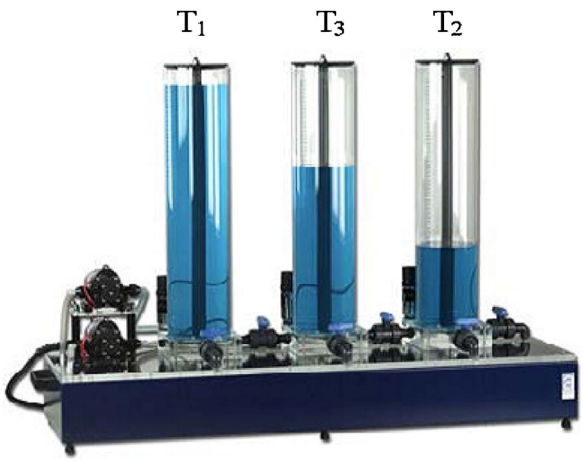

Fig. 3. Three-tank system.

cross-section, supplied with water. The liquid levels h_1, h_2 and h_3 are measured by piezoresistive transducers in the range of $[-10, +10]$ V. The middle tank (T_3) is connected to the other two adjacent tanks through circular cross-section pipes, provided with manually adjustable ball valves, while the main outlet of the system is located at tank T_2. This tank is connected to the bottom reservoir by means of a circular cross-section pipe provided with an outflow ball valve.

The WSN infrastructure relies on Zolertia Z1 sensor nodes, which leverage several industry standards, such as USB, IEEE 802.15.4 and Zigbee to interoperate seamlessly with other devices. The Z1 is a low power wireless device that comes with built in support for a number of popular open source operating systems, such as the TinyOS and Contiki, while the supported network stacks include 6LowPAN and Zigbee. Each node includes analogue and digital ports, to which sensors/transmitters can be attached.

The operating system used in WSN programming is based on the Contiki. This operating system has been written in C language with support for dynamic loading and replacement of individual programs and services. Additionally, it was built around an event-driven kernel, but provides optional preemptive multi-threading, which can be applied to individual processes (see e.g. [9,24]).

Regarding the configuration of the test-bed (Fig. 4), three nodes are configured as sensors, to which the level transmitters are connected in order to collect the tanks(tm) levels, h_1, h_2 and h_3, while the fourth node is used as a sink. The sink node is attached through a USB interface to the remote desktop computer (PC1) where the Tunslip is running, while the monitoring system is located on the remote desktop computer PC2. Finally, it should be recalled that the detection and accommodation algorithms are implemented under a multi-agent framework.

6.2. Results

In the following, the algorithms under consideration are simultaneously evaluated on the underlying sensor node, considering

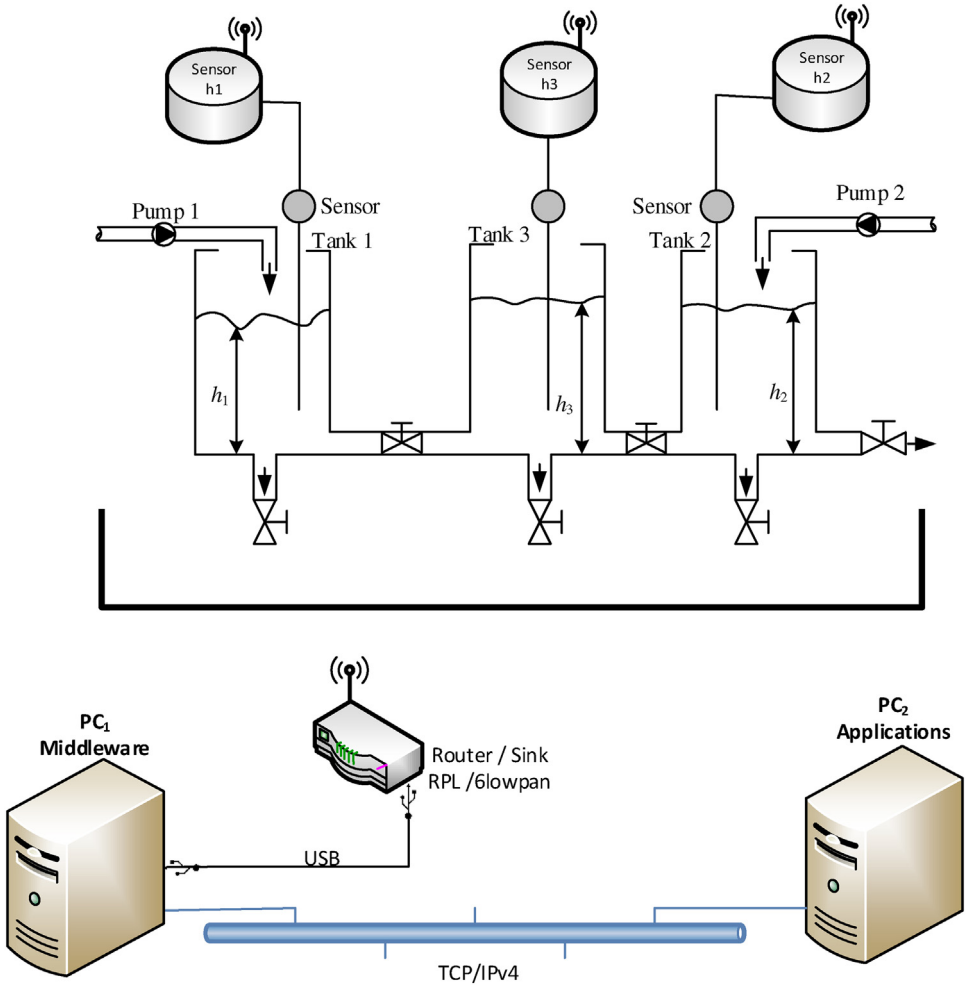

Fig. 4. Test-bed schematics.

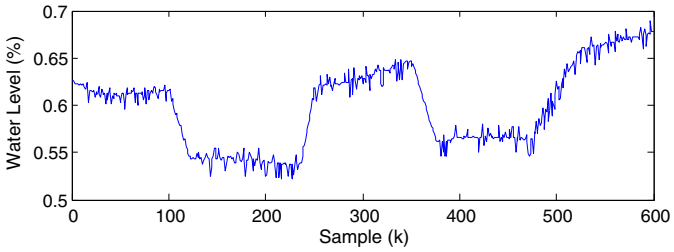

Fig. 5. Time-series collected from tank T_1.

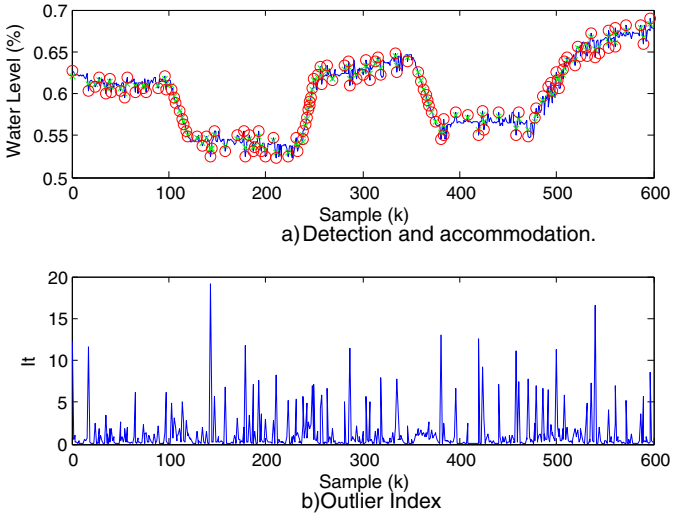

Fig. 6. SVND approach with the original kernel. (For interpretation of the references to colour in the text, the reader is referred to the web version of the article.)

readings taken from Tank T_1. The whole raw time-series after the experiment was completed is presented in Fig. 5.

The outcomes provided by the three methodologies are presented in Fig. 6, for the original SVND approach, Fig. 7 with regard to the SVND algorithm with the modified Gaussian kernel, in Fig. 8 for the R-OPASTr algorithm, and in Fig. 9 for the univariate statistical approach. In these figures tagged outliers are shown as red circles, while accommodated samples are represented with green crosses.

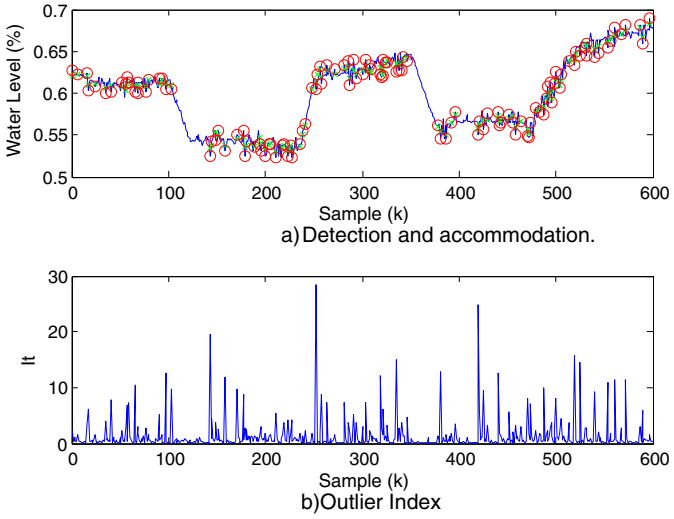

Fig. 7. SVND approach with the modified kernel. (For interpretation of the references to colour in the text, the reader is referred to the web version of the article.)

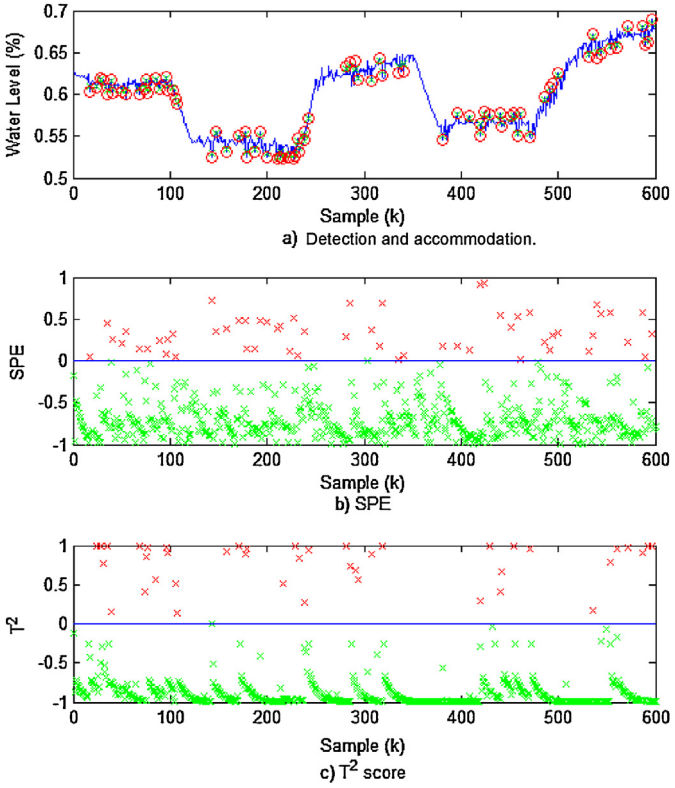

Fig. 8. R-OPASTr algorithm. (For interpretation of the references to colour in the text, the reader is referred to the web version of the article.)

As can be observed from Fig. 6, when the water level suddenly changes (transient response) the detection performance associated with the standard kernel-based approach deteriorates, which is due to the transient system's response, resulting in a I_t above the threshold. As such, in the case of transient time-series, the algorithm ends up tagging a significant number of samples as outliers. When applying the modified kernel-based algorithm, the underlying performance is notably superior in the region where the system's level undergoes a major change (see Fig. 7), while in the case of quasi-steady state operating regimes, the number of tagged outliers is rather higher. As for the R-OPASTr algorithm, Fig. 8 shows a better

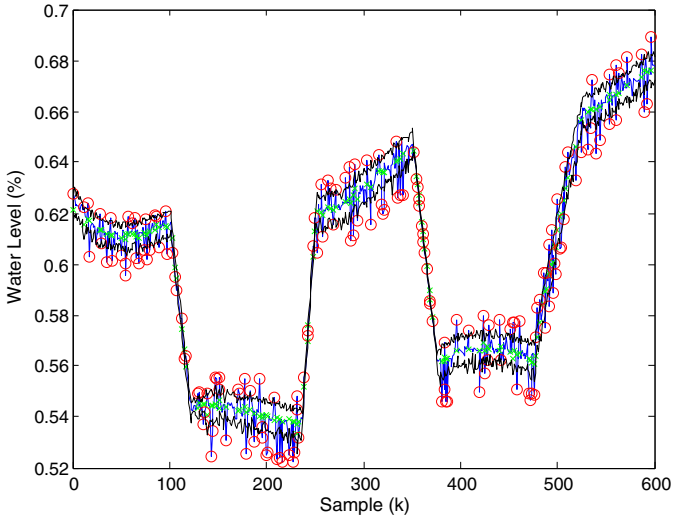

Fig. 9. Univariate statistical approach. (For interpretation of the references to colour in the text, the reader is referred to the web version of the article.)

Table 2
Comparing metrics.

Detection method	Original SVND	Modified SVND	OPAST-r algorithm	Univariate statistics
Total number of samples	600	600	600	600
Outlying samples	121	118	78	164
Detection rate	20.17%	19.67%	13.00%	27.33%
Minimum accom. ($\times 10^{-3}$)	−10.87%	−11.37%	−9.96%	−20.02
Maximum accom. ($\times 10^{-3}$)	11.19%	11.84%	9.67%	14.65
Accom. mean ($\times 10^{-4}$)	−5.37%	−7.93%	−7.68%	−5.77
Accom. std. deviation ($\times 10^{-3}$)	5.54%	6.24%	4.10%	5.41

performance than that provided by original kernel-based SVND in the transient operating regime, while in the case of quasi-steady readings it tags fewer outliers, which might impact its sensitivity. With regard to the univariate statistical-based approach, the number of samples tagged as outliers (see Fig. 9) is larger than that associated with the other approaches.

The results are summarised in Table 2, which compares the approaches under assessment in terms of detection rate, expressed as the relative value between the number of detected outliers and the total samples, along with some statistical measures regarding the outliers' accommodation.

7. Conclusions

The present paper evaluated three online outliers detection techniques with potential application in Wireless Sensor Networks. The first approach regards a machine learning technique based on a Least Squares-Support Vector Machine algorithm, under the form of a Reproducing Kernel Hilbert Space (RKHS) with Radial Basis Function (RBF) kernel, along with a sliding window-based learning technique. Aiming at improving the performance of this method in non-stationary conditions, a modification to the RBF kernel was suggested. It is characterised by replacing the Euclidean norm between adjacent readings with the norm of the corresponding differences to Least Squares estimates. The third methodology is a statistical PCA-based technique, with recursive subspace tracking and rank-1 modification, while the last approach relies on univariate statistics, along with an oversampling mechanism within each sampling interval. These methodologies were implemented on local sensor nodes within a hierarchical multi-agent framework, and assessed on a test-bed consisting of a WSN, a benchmark three-tank system and two desktop computers. Experiments have shown the feasibility of implementation of the tested methodologies on the WSN nodes and the superior performance, in terms of sensitivity and specificity, presented by the LS-SVM based approach with modified RBF kernel.

Acknowledgement

This work has been partially supported by Project CENTRO-07-ST24-FEDER-002003 (iCIS-Intelligent Computing in the Internet of Services).

References

[1] M.S. Bartlett, An inverse matrix adjustment arising in discriminant analysis, Ann. Math. Stat. 10 (1951) 7–111.
[2] J.W. Branch, C. Giannella, B. Szymanski, R. Wolff, H. Kargupta, In-network outlier detection in wireless sensor networks, Knowl. Inf. Syst. 34 (1) (2013) 23–54.
[3] D. Carels, N. Derdaele, E.De. Poorter, W. Vandenberghe, I. Moerman, P. Demeester, Support of multiple sinks via a virtual root for the RPL routing protocol, EURASIP J. Wirel. Commun. Netw. 2014 (1) (2014).
[4] S.-C. Chan, H.C. Wu, K.M. Tsui, Robust recursive eigendecomposition and subspace-based algorithms with application to fault detection in wireless sensor networks, IEEE Trans. Instrum. Meas. 61 (6) (2012) 1703–1718.
[5] L.H. Chiang, R.D. Braatz, E.L. Russell, Fault Detection and Diagnosis in Industrial Systems, Springer Science & Business Media, 2001.
[6] S. Chouvardas, Y. Kopsinis, S. Theodoridis, Robust subspace tracking with missing entries: the set-theoretic approach, IEEE Trans. Signal Process. 63 (19) (2015) 5060–5070.
[7] M. Davy, F. Desobry, A. Gretton, C. Doncarli, An online support vector machine for abnormal events detection, Signal Process. 86 (8) (2006) 2009–2025 (Special section: Advances in Signal Processing-assisted Cross-layer Designs).
[8] F. Desobry, M. Davy, C. Doncarli, An online kernel change detection algorithm, IEEE Trans. Signal Process. 53 (8) (2005) 2961–2974.
[9] A. Dunkels, Chapter: Operating Systems for Wireless Embedded Devices The Wiley Encyclopedia of Computer Science and Engineering, vol. 4, Wiley, Hoboken, NJ, USA, 2009, pp. 2039–2045.
[10] P. Gil, A. Santos, A. Cardoso, Dealing with outliers in wireless sensor networks: an oil refinery application, IEEE Trans. Control Syst. Technol. 22 (4) (2014) 1589–1596.
[11] M. Govindarajan, V. Abinaya, An outlier detection approach with data mining in wireless sensor network, Int. J. Curr. Eng. Technol. 4 (2) (2014) 929–932.
[12] A. Gualtierotti, Image Processing, Analysis, and Machine Vision, Springer, 2015.
[13] M. Ha, C. Wang, J. Chen, The support vector machine based on intuitionistic fuzzy number and kernel function, Soft Comput. 17 (4) (2013) 635–641.
[14] D.M. Hawkins, Q. Wu, The CUSUM and the EWMA head-to-head, Qual. Eng. 26 (2) (2014) 215–222.
[15] I.T. Jolliffe, Principal Component Analysis, Springer, 2002.
[16] H.F. Kaiser, The application of electronic computers to factor analysis, Educ. Psychol. Meas. 20 (1) (1960) 141–151.
[17] G. Li, C. Wen, Z.G. Li, A. Zhang, F. Yang, K. Mao, Model-based online learning with kernels, IEEE Trans. Neural Netw. Learn. Syst. 24 (3) (2013) 356–369.
[18] M.A. Mahmood, W.K.G. Seah, I. Welch, Reliability in wireless sensor networks: a survey and challenges ahead, Comput. Netw. 79 (2015) 166–187.
[19] A. Patel, Control Schemes for an Analytical Process Data: Modified Univariate and Multivariate Exponentially Weighted Moving Average Control Schemes, LAP LAMBERT Academic Publishing, 2012.
[20] W. Pratt, Introduction to Digital Image Processing, CRC Press, 2013.
[21] M.A. Rassam, A. Zainal, M.A. Maarof, Advancements of data anomaly detection research in wireless sensor networks: a survey and open issues, Sensors 13 (8) (2013) 10087–10122.
[22] P. Rawat, K.D. Singh, H. Chaouchi, J.M. Bonnin, Wireless sensor networks: a survey on recent developments and potential synergies, J. Supercomput. 68 (1) (2014) 1–48.
[23] D. Tuia, J. Munoz-Mari, J.L. Rojo-Alvarez, M. Martinez-Ramon, G. Camps-Valls, Explicit recursive and adaptive filtering in reproducing kernel Hilbert spaces, IEEE Trans. Neural Netw. Learn. Syst. 25 (7) (2014) 1413–1419.
[24] J.-P. Vasseur, A. Dunkels, Interconnecting Smart Objects with IP - The Next Internet, Morgan Kaufmann, 2010.
[25] M. Xie, S. Han, B. Tian, S. Parvin, Anomaly detection in wireless sensor networks: a survey, J. Netw. Comput. Appl. 34 (4) (2011) 1302–1325.
[26] Y. Zhang, N. Meratnia, P. Havinga, Outlier detection techniques for wireless sensor networks: a survey, IEEE Commun. Surv. Tutor. 12 (2) (2010) 159–170.
[27] Y. Zhang, N.A.S. Hamm, N. Meratnia, A. Stein, M. van de Voort, P.J.M. Havinga, Statistics-based outlier detection for wireless sensor networks, Int. J. Geogr. Inf. Sci. 26 (8) (2012) 1373–1392.
[28] Y. Zhang, N. Meratnia, P.J.M. Havinga, Distributed online outlier detection in wireless sensor networks using ellipsoidal support vector machine, Ad hoc Netw. 11 (3) (2013) 1062–1074.