A Portable Electronic Nose Based on Embedded PC Technology and GNU/Linux: Hardware, Software and Applications

Alexandre Perera, Teodor Sundic, Antonio Pardo, Ricardo Gutierrez-Osuna, *Member, IEEE*, and Santiago Marco

***Abstract*—This paper describes a portable electronic nose based on embedded PC technology. The instrument combines a small footprint with the versatility offered by embedded technology in terms of software development and digital communications services. A summary of the proposed hardware and software solutions is provided with an emphasis on data processing. Data evaluation procedures available in the instrument include automatic feature selection by means of SFFS, feature extraction with linear discriminant analysis (LDA) and principal component analysis (PCA), multi-component analysis with partial least squares (PLS) and classification through k-NN and Gaussian mixture models. In terms of instrumentation, the instrument makes use of temperature modulation to improve the selectivity of commercial metal oxide gas sensors. Field applications of the instrument, including experimental results, are also presented.**

***Index Terms*—Distributed sensing, electronic nose, embedded, ipNose, linux, mixture models, SFFS, signal processing, smart.**

I. Introduction

EARLY electronic nose prototypes were complex and rather bulky laboratory systems. Soon after their first appearance in the marketplace, these instruments underwent a first step toward size reduction and desktop systems appeared [1]. Nonetheless, the inclusion of automatic headspace samplers and a controlling PC made these instruments only suitable for laboratory purposes. Lately, the trend toward miniaturization has lead to the appearance of handheld instruments, like the VOCCheck® from AppliedSensors(Germany/Norway), VaporLab® from Microsensor Systems Inc. (USA), or Cyranonose® from CyranoSciences Inc. (USA). These remarkably small-sized instruments feature simple gas sampling procedures and rely on microcontrollers/microprocessors for instrument control and data processing. However, their versatility and potential applications are reduced when compared to their desktop counterparts. While it is clear that sampling procedures in field-portable instruments have to be simple, all the other capabilities of a smart portable instrument should be similar to their desktop versions.

It is the purpose of this paper to present a portable electronic nose based on embedded PC technology and the Linux operating system. While this instrument still features simple headspace sampling, our work shows that many smart features can be included in a small-sized instrument taking advantage of state-of-the-art embedded technology. This paper is divided as follows. Section II introduces system design considerations regarding the implementation of pattern recognition procedures compatible with system resources and a detailed overview of the different hardware components of the instrument. Section III focuses on the signal processing algorithms currently available in the system software. Section IV presents an experimental validation of the proposed techniques as well as an application of the instrument for environmental monitoring.

Manuscript received August 12, 2001; revised May 20, 2002. This work has supported in part by Generalitat de Catalunya ACI2001-11 and Ministerio de Ciencia y Tecnología DPI2001-3213-C02-01. The associate editor coordinating the review of this paper and approving it for publication was Dr. Krishna Persaud.

A. Perera, T. Sundic, A. Pardo, and S. Marco are with the Sistemes d'Instrumentació i Communicacions, Departament d'Electrònica, Universitat de Barcelona, Barcelona, Spain (e-mail: aperera@el.ub.es; teodor@el.ub.es; toni@el.ub.es; santi@el.ub.es).

R. Gutierrez-Osuna is with the Department of Computer Science, Wright-State University, Dayton 45435 OH USA (e-mail: rgutier@cs.wright.edu).

Publisher Item Identifier 10.1109/JSEN.2002.800683.

II. Electronic Nose System Design

A. Computing Considerations

The usual definition of an electronic nose states that the instrument includes nonspecific sensors plus a pattern recognition system [2]. When designing the intelligent processing and smart operation components of an electronic nose, several approaches that range from powerful desktop systems to portable systems with limited computational resources should be considered depending on the required instrument size and final application

The classical structure of systems of this kind is shown in Fig. 1. When designing embedded pattern analysis software, it is important to keep in mind that the complexity of training (or estimation) algorithms tends to be much higher than the complexity of the algorithms in the operation phase. A typical example could be principal component regression (PCR), where simple matrix manipulation is needed in the operation phase, but more complex routines such as singular value Decomposition (SVD) may be needed during model estimation. For platforms with limited computing power, some algorithms should be trained off-line, usually in a host computer and parameters delivered to the system using appropriate digital communications. For more powerful platforms, the same instrument will be able to adapt their data processing scheme to the problem of interest. If some learning engine has to be implemented in a portable device, computational resources are an issue to be taken in account in the system design, since there will be limitations in speed and often in memory. Some alternatives are presented in Table I.

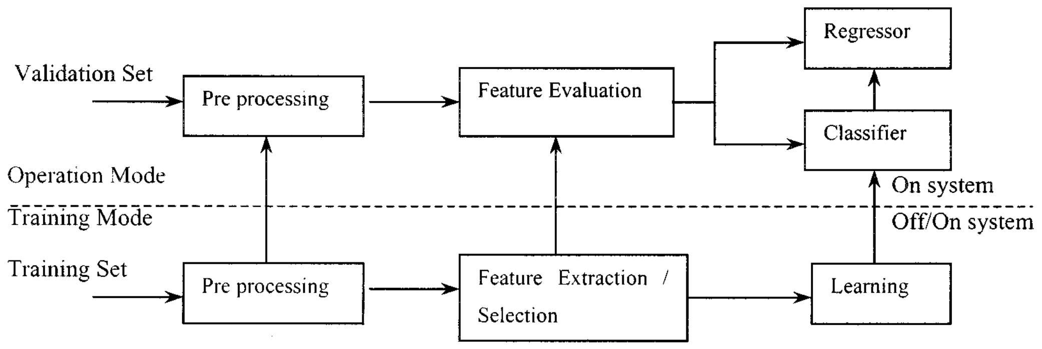


Fig. 1. Typical architecture of signal processing. Validation or normal mode is always computed in the instrument but training or learning mode is usually performed in a computer.

TABLE I
ARCHITECTURAL ALTERNATIVES IN THE DESING OF AN ELECTRONIC NOSE

Architecture	*Typ. Conf. (bits / speed / RAM)*	*Pros/cons*	*Typ. Programmin g Lang.*	*Available Processing*
Sensor Array + µC (PIC)	8 bit / 10 MHz / kbytes	Easy, small, low power, Portable, Cheaper	ASM / C	Easy algorithms with few data, k-NN, easy NN, mostly trained off-system. Linear classifiers, quadratic classifiers
SA + high perf. MC	8-16 bit / 10-33 MHz / kbytes	Small, low power, Portable, Cheap	ASM / C	Some small matrix manipulation available, Linear (PCA, LDA, PCR), k-NN. Easy Fuzzy Inference Systems
SA + µP or DSP	16-32 bit / 20-100 MHz / Mb	Very Fast, medium size, Portable High consume	ASM / C / C++	Linear, (PCA, LDA, PCR, PLS), k-NN, easy Neural and Fuzzy Systems. Standard Feature Extraction/Selection (PCA, LDA)
SA + Embedded PC	32 bit / 80-233 MHz / Mb	Fast, medium size, Portable, Huge data capacity, High consume Expensive	Any	Linear, Complex Learning algorithms (GA, NeuroFuzzy Systems, Mixture Models, APR, FIS Optimization Algorithms). Advanced Feature Extraction/Selection (SFS, SFFS)
SA + Desktop PC	32-64 bit / 700MHz /Mb	Fast, medium size, Huge data capacity, Consume not critical, Expensive, Not portable	Any/ Visual	Linear, Complex Learning algorithms (GA, NeuroFuzzy Systems, Mixture Models, FIS Optimization Algorithms). Advanced Feature Extraction /Selection (SFS, SFFS ...), etc

In order to develop the e-nose software, some assumption should be made about the hardware solution (i.e., if dynamic memory is available). In any case, to maximize software portability between different platforms, we advocate for a modular system that can be customized at compilation time. This will allow the solution to become portable among different architectures in a range of instrument designs for an e-nose manufacturer.

The use of embedded technology provides several interesting benefits: availability of an abstraction layer for signal acquisition and control via an operating system, high level programming of the signal processing algorithms, large data storage in solid state disks, commercial-off-the-shelf hardware for Ethernet connectivity, I^2C and CAN buses, serial ports, hardware for interfacing various types of displays, etc. Furthermore, an impressive trend to reduce cost and size of this kind of systems can be observed in the market place.

B. IpNose Electronic Nose Description

In this paper, we propose the use of embedded PC technology with the GNU/Linux operating system and appropriate pattern recognition/regression software in our ipNose electronic nose prototype. While this instrument still features simple headspace sampling, our work shows that many smart features can be included in a small-sized instrument taking advantage of state-of-the-art embedded technology.

An overview of the design is shown in Fig. 2. The instrument can control up to three sensor modules. Each module consists of four metal oxide sensors, one temperature sensor and signal conditioning/excitation electronics on a custom printed circuit board (PCB). The electronics can interface various commercial sensors, including FIS (Japan), FIGARO (Japan), MICROSENS (Switzerland), MICS(Switzerland), or Capteur Sensors Ltd (UK) via configuration jumpers, although in the

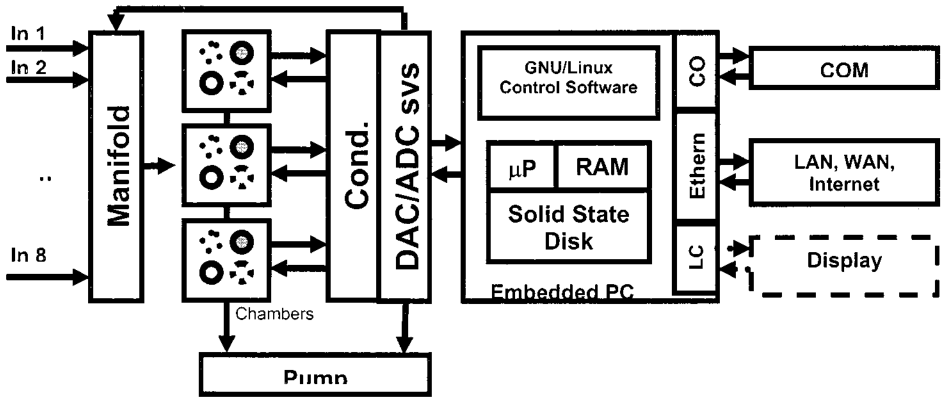


Fig. 2. ipNose structure.

current configuration only four SB-series FIS sensors are used. These sensors (SB-95-11, SB-31-00, SB-AQ1A-00 & SB-11A-00) present an internal structure based on a micro-bead of sensing material deposited over a coil. This structure provides the sensors with a fast thermal response to a modulating heater voltage. The instrument design, both hardware and software, is aimed at allowing sensor parameter modulation for better aroma/gas discrimination.

The flow injection system consists of an eight-channel manifold with corresponding miniature electro-valves for each intake port. The software controls both the order and the aperture time of each valve and the pump, as defined in a configuration file. A reference channel, including a zero filter for air reference, is also available. The output of the manifold connects directly to the sensor chamber. The sampling system operates by means of a miniature pump connected downstream from the sensor chamber. A check valve is placed between the chamber and the pump to prevent backflow.

Our implementation uses an embedded computer system based on the PC/104 standard [3]. This consortium defines compact size self-stacking modules (9 cm $\times$ 9.5 cm) and a PCI bus across the stack. In our instrument, the stack is composed by four elements: a 16-channel input and a 2-channel output data-acquisition converter, an eight-channel relay board, a dc-dc converter and a single board computer (SBC). The data acquisition module interfaces with signal conditioning electronics for sensor signal acquisition and heaters excitation. The relay module controls the valves array and the pump. The dc-dc converter distributes power to the system using a single 6–40 V input supply (e.g., a battery). Finally, the SBC houses a low-power GEODE processor from National Semiconductor Inc., 32 Mbytes RAM memory, interface circuitry (COM ports, USB and Ethernet) and includes a 64 Mb CompactFlash® solid-state disk where the operating system is stored.

Various operating systems are available to run in these platforms, including Microsoft NT® embedded, Windows CE, or open source UNIX-clone operating systems (e.g., FreeBSD, GNU/Linux), some of them are already being used for instrumentation purposes [4]. The open-source option has been selected due to the availability of the source code, which allows the developer to customize the operating system to meet hardware constraints. The core of the system is an embedded computer running an embedded GNU/Linux operating system. The use of high-end computing hardware allows complex multivariate analysis of high-dimensional patterns such as those typical of temperature-modulated metal–oxide sensors [5]. Although the default temperature modulation is a square wave (as described in the next section), arbitrary temperature profiles can be easily programmed or uploaded into the system as text files (e.g., generated with MATLAB). The use of solid-state drives allows the system to be used as a portable intelligent volatile detector, as well as a data-acquisition instrument for further processing in laboratory.

The instrument also provides the means for performing distributed multi-instrument sensing, taking advantage of the connectivity capabilities that GNU/Linux provides. The idea of providing remote connectivity to an electronic nose is geared toward expanding the usability of these instruments for industrial applications and distributed sensing. The ipNose design provides a re-programmable platform for remote operation. It performs scheduled or periodic analysis and transmits results via a phone call to a host computer or to an Internet site via an Ethernet device or a Point to Point Protocol (ppp) connection. It receives incoming calls, allowing the operator to upload scheduling tables or download acquisition data from the instrument. Incoming remote connections are processed with the help of a background process that serves signals and analysis results in the same fashion as web servers.

Once a connection is established, commands can be sent to the instrument to perform sampling or training, obtain current sensors values, change heater voltages, control pump and valves, or even re-program the instrument. As illustrated in Fig. 3, these features allow the user to perform log analysis, extract or modify internal database parameters, or change signal processing software of a distributed system of ipNoses from any workstation. Although the e-nose system is remotely operated via TCP/IP under client/server architecture, it can also be configured to send active signals to external systems (e.g., e-mail the user when samples do not match specifications). The size of the instrument is 30 $\times$ 15 $\times$ 15 cm. A picture of the instrument is shown in Fig. 4.

III. IPNOSE SIGNAL PROCESSING

A complete library for signal filtering, signal pre-processing, feature selection, matrix manipulation, pattern recognition, and

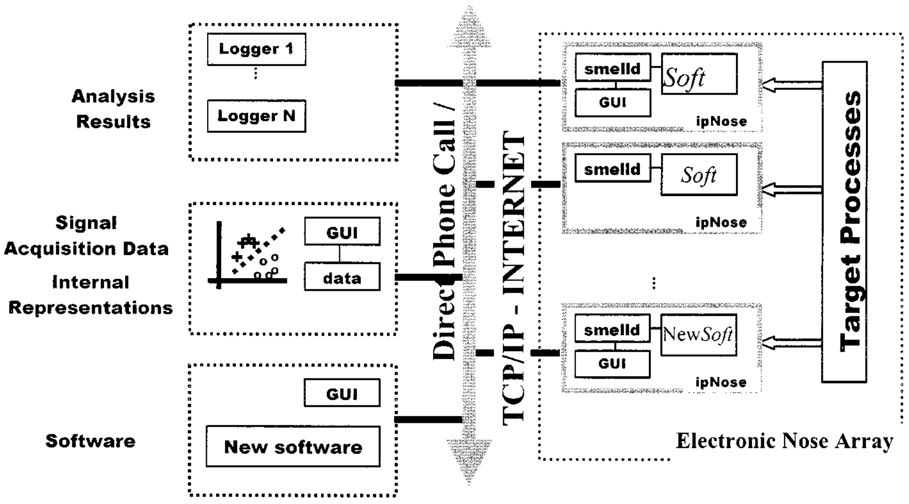


Fig. 3. Remote connectivity features diagram for ipNose.

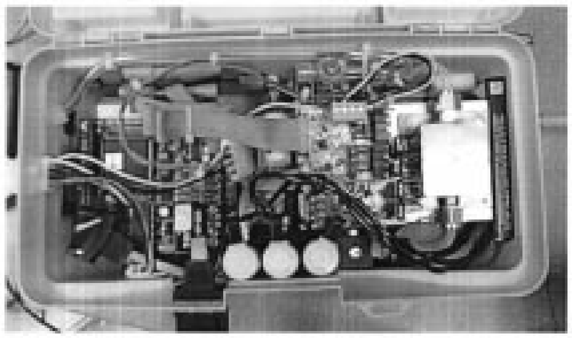


Fig. 4. Picture of ipNose.

pattern regression is contained in the instrument. Linear discriminant analysis (LDA), principal component analysis (PCA), and partial least squares (PLS), as well as k-nn and Gaussian mixture models classifiers, are currently included in the instrument's signal processing software toolbox. In the present work, we will focus our description on the pre-processing, feature selection, and mixture model classifier components. We will illustrate the different processing stages with experimental electronic nose data.

A. Signal-Processing

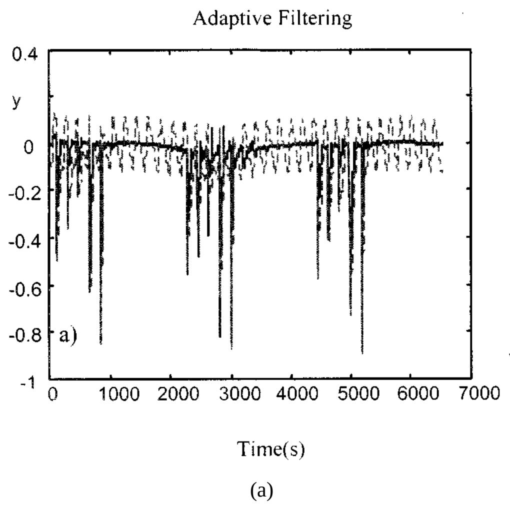
Although signal processing will depend on the application, a series of steps are commonly carried out. Most e-noses use as raw signals the transients that appear when commuting from the reference line to the analysis line. For many and diverse reasons, these signals appear contaminated with broad-band intrinsic noise, narrow-band noise from electronic interference, and cross-sensitivities to ambient humidity and temperature changes. Moreover, outliers are not uncommon. All these perturbations contribute to cluster scattering and they are an ultimate limitation to system performance apart from the pervasive sensor drift. With improved signal processing power, real-time digital filtering becomes an attractive possibility [6].

The ipNose uses a Savitzky–Golay polynomial filter for broadband nose removal. Savitzky–Golay (SG) polynomials are low pass filters that provide maximum noise rejection, with minimum transients and minimal distortion of low frequency signals. SG filters are optimal when the signal can be locally approximated by a polynomial of degree p [7].

Sometimes, the perturbation can be periodic. A typical case are conductive interferences from pulsed sensors on board. In this case, comb notch filters are a good option as they can eliminate the fundamental frequency and related harmonics. Such a filter provides perfect rejection and easy control of the notch bandwidth.

In some cases, it is important to cancel interferences of a known origin. Temperature and humidity variations are common problems for electronic noses in field applications [8]. We encountered this problem while operating the electronic nose inside a refrigerated chamber. In this environment, large humidity excursions are caused by the temperature control. This humidity drives the sensor signals with amplitudes that can be as large as the signals of interest. However, this humidity interference can be compensated if the interferences are simultaneously measured. In this particular, case we had access to the relative humidity inside the refrigerator by means of an independent capacitive humidity sensor. In these cases, static compensation is often not optimum because of the different time response between sensor array and humidity sensor. In these situations, adaptive filters are the best option as they adapt to the different time behavior of both sensors. Fig. 5 shows the noise rejection obtained with an adaptive filter (7-taps) trained with the LMS algorithm [9]. The operation of this kind of filter is described in Fig. 6. The filter coefficients adapt following the LMS rule minimizing the power of the error signal. This calibration signal is available when the sensor array is measuring the reference line. At this time, it is assumed that most of the periodic variations can be attributed to the interference to be removed.

Outlier detection can be carried out basically through the Hotteling T2 statistics and the residual power in a PCA analysis. However, a preventive simple median filter provides good results in most cases where outliers are due to spikes. These nonlinear filters are much more efficient at removing spike noise



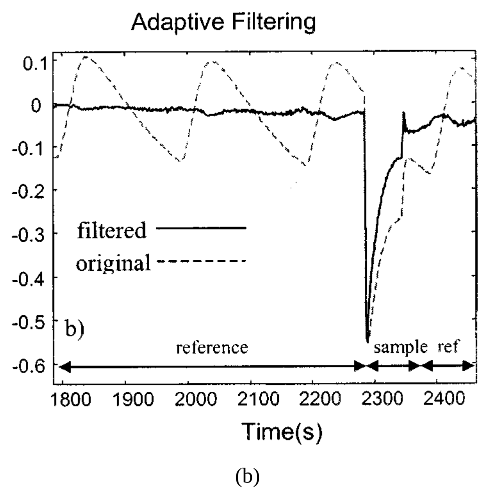


Fig. 5. a) Effect of LMS adaptive filtering for a sensor; b) detail showing the retrieval of baseline and an resulting signal when switching to a sample channel.

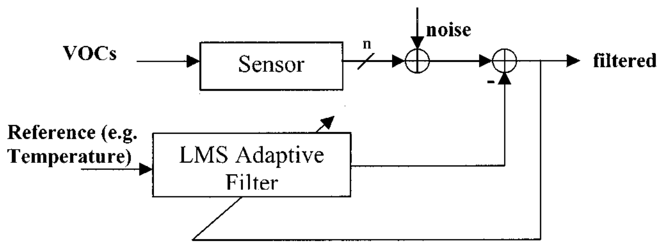


Fig. 6. LMS adaptive filtering description.

than linear filters and are used routinely in image processing (i.e., for salt-and-pepper noise removal).

B. Feature Extraction

As a second step features are extracted from the signal. For dc heating, generated feature vectors tend to be of low dimensionality. Several options are reported in the literature for this situation [2], for instance

$$v_i = v^{\max} - v^{\min} \; v_i = \frac{v^{\max} - v^{\min}}{v^{\min}}$$
$$v_i = \frac{v^{\max}}{v_{\min}} \; v_i = \log\left(\frac{v^{\max}}{v_{\min}}\right) \tag{1}$$

where v is the conductance or resistance of the sensors. However, sometimes additional information may be extracted from the signal transients. In this case, and also when using temperature modulation, most sampled measurements contains redundant information. It then becomes mandatory to reduce the effective sampling frequency by reducing real sampling frequency or by perform a time-signal decimation in order to avoid dimensionality problems. For classification purposes, it is customary to include a data normalization step (across the sensor array) in order to reduce the pattern dispersion induced by concentration changes [10]. Finally, every feature is scaled to zero mean and unity variance to compensate for differences in the dynamic range.

Even after sub-sampling, the resulting patterns are still of high dimensionality. Due to the scarcity of data and the corresponding curse of dimensionality, it is usual to proceed with a data reduction step by means of feature extraction or feature selection. Among the various feature extraction techniques, PCA and LDA are the default option in our electronic nose. Usually, PCA is carried out on the raw signal transient followed by supervised LDA. Other approaches, such as self organizing maps or non linear PCA, are currently being considered [11].

C. Feature Selection

The reasons for reducing the number of input variables are twofold. First, by eliminating unimportant sensors or extracted features, the cost and time of processing the data can be reduced. Second, the performance of the model can be improved by discarding features containing irrelevant information, reducing the co-linearity of the feature vector in addition to requiring a lower number of training examples. The optimal solution may be found by an exhaustive search. The principal setback of this method is the exponential growth of all possible subsets with the total number of features, which makes it unpractical even for a moderate number of features, so other search algorithms must be applied. Usually, these algorithms do not guarantee an optimal set of features, so they are sub-optimal.

The problem of feature selection can be defined as that of selecting a subset of features that performs best in a particular classification/prediction problem. More specifically, let $\boldsymbol{X}$ be the original data set containing p different features and n measurements. The objective is to find a subset $\boldsymbol{Z}(\boldsymbol{Z} \subseteq \boldsymbol{X})$ containing d features that minimizes a selection criterion $J(Z)$ [12]. The selection criterion is the measure used to rank feature subsets. Depending on the target problem various criteria can be used. If our problem focuses in classifying different samples, then classification rate is used. When feature qualities are predicted (e.g., concentration), regression error is then used.

There exist a certain number of sequential methods for feature selection. Sequential forward selection (SFS), which is the simplest one, adds one variable at a time to the model. The procedure begins by considering each of the features individually and selecting the feature $z1$ that gives the lowest value for the selection criterion J. The next step is then to calculate all possible two features subsets that include $z1$. Again, the feature that gives the lowest J is added to the subset. This process continues until J increases with the addition of any new variable or all features are included in the model. The main drawback is that once a variable has been added it cannot be removed. The backward elimination method operates in the opposite direction. In this case, all p variables are included in the beginning. Variables are then removed one at a time based on their contribution to the selection criterion J. There exist several variations of the forward selection and backward elimination schemes, like stepwise regression, *Plus-l-Minus-r* algorithm, etc.

The way in which these algorithms are applied in the ipNose depends on the characteristics of the data. The selection algorithms can be used for sensor selection by grouping all features for each sensor. Alternatively, particular temperatures can be selected over the signal waveform in temperature modulated sensors.

In *floating* selection methods [13], [14], which have been applied to the ipNose, the number of added and removed features at each selection step is adaptive. This means that, for each step, the algorithm searches both forward and backward (adding or removing one feature) in search for the combination that gives the lowest prediction error. By keeping flexible the number of added and removed features, there is a better chance at avoiding local minima. One disadvantage, though, is the longer processing time compared to standard forward or backward selection. Apart from the sequential feature selection methods, a few other approaches have also been reported in literature. In particular, genetic algorithms have been proved to perform well in many applications [15], [16], although the computational effort is higher than for SFFS while providing similar performance. The performance of the feature extraction/selection and pre-processing techniques is tested with a classical nearest neighbor classifier algorithm (k-NN), which does not require to train a model and presents minimum computational effort for small databases.

In ipNose, once appropriate pre-processing and feature extraction/selection procedures have been determined, these signal-processing meta-parameters are stored in a database definition that specifies the operation mode of the system for normal operation. The definition database also includes the final classifier to be used along with its learned parameters. Two classification approaches are currently available in the ipNose: k-NN voting and Gaussian mixture models.

D. Classification With Gaussian Mixture Models

k-NN is a nonparametric technique that provides good performance when used in conjunction with a previous stage of PCA-LDA, but also has important drawbacks (e.g it is memory demanding and sensitive to feature scaling). Parametric classifiers, on the other hand, attempt to build a model of the class-conditional probability densities in feature space. In particular, finite mixture models (FMM), estimate each class-conditional density using a mixture of simpler probability density functions. We first give an overview of how a given class-distribution is modeled by a mixture.The use of these class-distributionsas classifiers will be shown later on.

In a mixture model, the probability function of one particular class-distribution is considered as a linear combination of K components or kernels [17], [18] described as the so-called *mixture distribution*

$$p(x) = \sum_{i=1}^{K} \omega_i \cdot p(x|i) \tag{2}$$

where $p(x)$ is the probability density function of our dataset, $\omega_i = P(i)$ are the *prior probabilities* (also known as *mixing probabilities* or *mixing parameters*), and $p(x|i)$ are the posterior probabilities (probability that observation x was generated from mixture component i). This formulation is actually equivalent to the Bayes theorem.

The densities $p(x|i)$, referred to as likelihood functions, are parametric distributions for a given mixture component. In this paper, the individual probability density functions are assumed to follow a mixture of Gaussian distributions, hence the term Gaussian mixture models (GMM). Although the actual class-conditional distributions in feature space are generally non-Gaussian, the resulting multi-modal approximation is remarkably accurate. Each mixture component is defined by a normal parametric distribution in d dimensional space

$$p(x|i) = \frac{1}{(2\pi)^{d/2} |\Sigma_i|^{1/2}} \exp\left(\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)\right). \tag{3}$$

The parameters for each normal distribution component are the covariance matrix Σ_i, and the mean vector μ_i. The parameters to be estimated are defined as $\Theta = \{\theta_i; 1 \le i \le k\}$ where $\theta_i = \{\omega_i, \Sigma_i, \mu_i\}$. The total number of parameters to be calculated is, therefore, $n_p = c(1 + d(1 + (d+1)/2))$, where c is the number of components and d is the dimensionality of data.

Two methods are commonly used for training mixture models: the expectation-maximization (EM) algorithm [19] and Markov-Chain MonteCarlo (MCMC) methods [20]. In this work, we use the expectation maximization algorithm as MCMC would need too computing resources for on-system training. Briefly, EM produces maximum likelihood estimates of distribution parameters Θ, where the log likelihood function is defined as

$$L(\Theta) = \ln(p_\Theta(x)). \tag{4}$$

Usually, as the name suggests, the EM algorithm consists of two stages: an expectation step followed by a maximization step. The two steps are iterated until a convergence criteria is reached.

E-Step: The expectation step computes the expected value of $\boldsymbol{x}$ data using *old* estimates of Θ parameters and *new* observed data

$$p^{old}(x) = \sum_{i=1}^{K} \omega_i^{old} \cdot p^{old}(x|i) \tag{5}$$

where, assuming normal distributions, $p^{old}(x|i)$ are a function of the old estimates Σ^{old} and μ^{old}.

M-Step: Where the parameter estimates for Θ are update so that (4) is maximized with respect to the previous iteration.

In the particular, case of Gaussian mixture models, it can be shown that the EM procedure can be reduced to a relatively simple set of equations [21], where the estimate for the means for each component is given by

$$\hat{\mu}_i^{new} = \frac{\sum_{s=1}^{m} p^{old}(i|x^s) \cdot x^s}{\sum_{s=1}^{m} p^{old}(i|x^s)} \tag{6}$$

and, by using these new mean estimates, the new variances are estimated by

$$\hat{\Sigma}_i^2 = \frac{1}{d} \frac{\sum_{s=1}^{m} p(i|x^s) \cdot (x^s - \hat{\mu}_i)^T \Sigma^{-1} (x^s - \hat{\mu}_i)}{\sum_{s=1}^{m} p(i|x^s)} \tag{7}$$

and, finally, the priors are re-estimated by

$$\hat{P}(i) = \frac{\sum_{s=1}^{m} p(i|x^s)}{m}. \tag{8}$$

The EM algorithm is therefore an iterative procedure easy to implement and fairly robust. There exist many modifications to his basic procedure such as changing the complexity of the model via self-annealing algorithms [22]. EM guarantees a monotonically nondecreasing likelihood [23] although its ability to find a local maximum depends on parameter initialization.

The main drawbacks of EM are a slow convergence and the computation of an inverse of the covariance matrix Σ, which can become singular under certain conditions. Convergence can be accelerated by modifications on the *maximization* step and singularities in the covariance matrix estimates can easily be resolved with the use of a regularization term. This last case is often found when training mixtures in high dimensionality spaces, as the number of parameters scales to d^2 and the mixture holds only one component.

The classifier used in this work learns the conditional probability of each distribution using the labeled training data [24]. In other words, a mixture model is constructed for each class. Once all parameter estimates are calculated, classification of a test sample is done by choosing the mixture that gives the maximum probability.

The initialization of the classifier is done using the following procedure: (i) build a subset $\boldsymbol{X}^l$ corresponding to the first l class, (ii) estimate the maximum number of components K_l of current l mixture using training data, (iii) choose K_l random subsets $\boldsymbol{X}_k^l$ and (iv) initialize the components using the following intuitive rules

$$\mu_k^l = \bar{X}_k^l \quad \Sigma_k^l = \text{cov}\left(\bar{X}_k^l\right) \quad \omega_k^l = \frac{1}{K_l} \quad \text{for } l \le k \le K_l. \tag{9}$$

The initial number of components K_l is chosen by means of a heuristic rule given by

$$K_l = 1 + \frac{\sqrt{samples\,(X^l)}}{d} \tag{10}$$

where d denotes the dimensionality of feature space. Expression (10) gives a reasonable number of training samples per component in the sense that the minimum number of samples is always greater than the dimensionality of the space. A descriptive plot is shown in Fig. 7. Notice that high dimensionality will force the algorithm to construct a mixture model with only one component. When this situation occurs, the classifier has the same behavior as a Bayesian quadratic classifier, where the decision boundaries are hyper-ellipsoids or hyper-paraboloids.

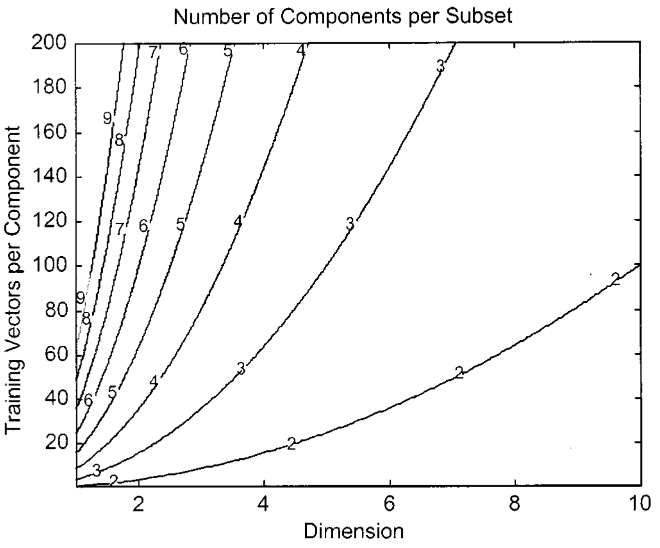


Fig. 7. Selection of number of components per mixture.

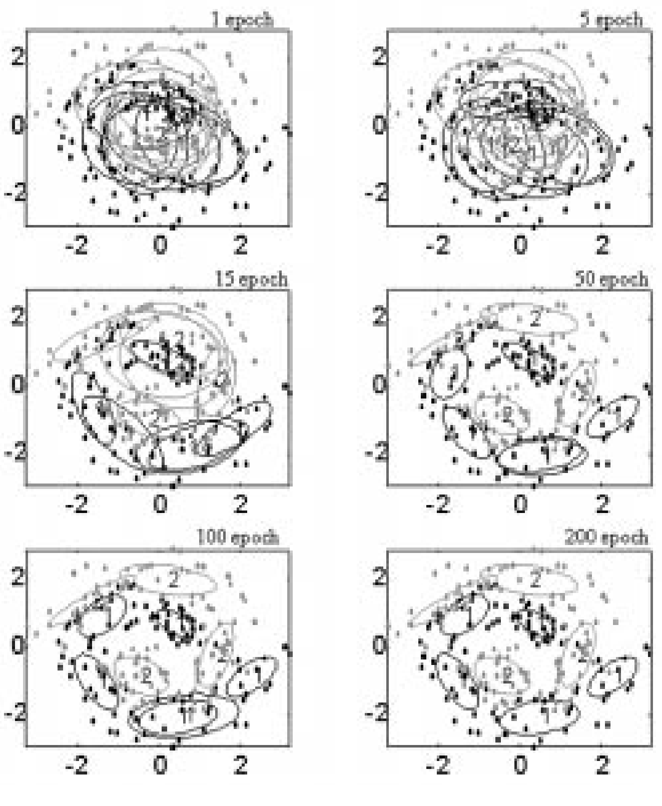
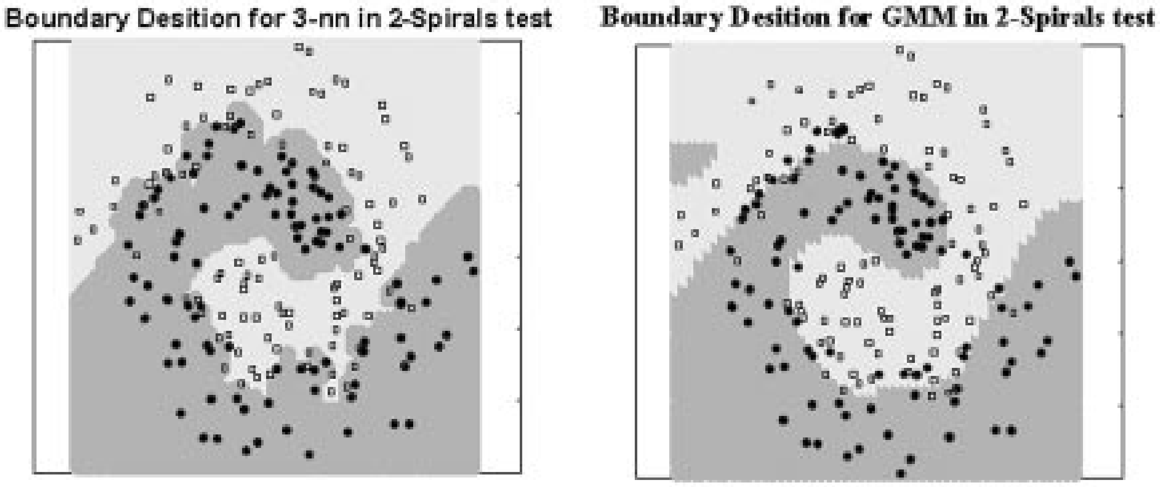


Fig. 8. Evolution of two-spiral classification synthetic test.

In order to illustrate the behavior of the classifier, we will employ a classification problem with highly nonlinear synthetic data. The data consists of two complementary spirals where each spiral corresponds to a class. There are 100 samples per

TABLE II
EIGHT-FOLD CONFUSION MATRIX FOR TWO-SPIRAL TEST. (*TR = TRAIN RATE, VR = VALIDATION RATE)

Predicted\Real	3-NN (*TR=83%, VR=78%)		GMM (TR=85%, VR=81%)	
Training Set	A	B	A	B
A	574	109	586	99
B	130	595	118	605
Validation Set	A	B	A	B
A	75	21	78	19
B	21	75	18	77



(a) (b)

Fig. 9. Boundary Decision for two-class spiral test. (a) k-nn Test, (b) GMM test.

class and classification rates will be estimated using eight-fold cross validation.

The evolution of the components is shown Fig. 8. Notice that some components are removed as the iterations progress. This is related to the fact that we allow priors to be adapted at each EM iteration. The annealing of a component is done when the prior becomes lower than a pre-specified threshold. This is based on the idea that a low prior component has little influence in the log-likelihood of the data. We can compare the classification ratio performance against a standard k-NN classfier. Results are shown using the confusion matrix of Table II.

We observe slightly better results for the mixture model both in terms of classification rate and boundary stability ((Validation Rate (*VR*))/(*Training Rate (TR)*)$=$ 0.94 for *K-nn* and $VR/TR = 0.95$ for GMM).Validation stability can also be observed in the boundary plots pictured in Figs. 9(a) and 9(b). As expected, the mixture model has produced a smoother contour than k*-NN*, the latter being characteristic of a small value of k compared to the number of samples per class. If we increase k, the classification ratio decreases dramatically for k*-NN*, as the classifier looses its strong locality properties that allow it to generate highly nonlinear decision boundaries.

Multicomponent regression procedures are also included in the ipNose. Classical methods like PCR [25], PLS [26] and a Fuzzy Inference System described in [27] are available to the user.

IV. EXPERIMENTAL

In the following section, we will show the overall behavior of signal processing stage when heuristic feature extraction and systematic feature extraction is performed over temperature modulated signals. We will compare the results using the same dataset.

The data set was collected using the ipNose instrument on three different days. The dataset consists of three different odors (coffee, tobacco, and cedar-wood) plus air. 16 samples were taken for each odor in random order. Sampling was not done under standard headspace analysis but rather by sniffing near the surface of the substance (within 3–7 cm). In this situation, concentration levels may (and did) have high variability.

The IpNose can be configured to perform different cycles with different active channels and arbitrary temperature modulation. For the experiments presented in this section, the instrument is configured to run three cycles while applying a periodic square waveform to the heater voltages. Each period is 10 s long. During the first three seconds, the sensor is excited with 0.9 V, reaching a 400 °C operating temperature. During the remaining seven seconds the heaters are set at 0.2 V, reaching 80 °C. A first cleaning cycle consisting of eight periods is used to ensure that the chamber is free of volatile compounds. In the second four-period cycle, odors are introduced into the chamber. The third cycle is a purging stage in which the system is flushed with

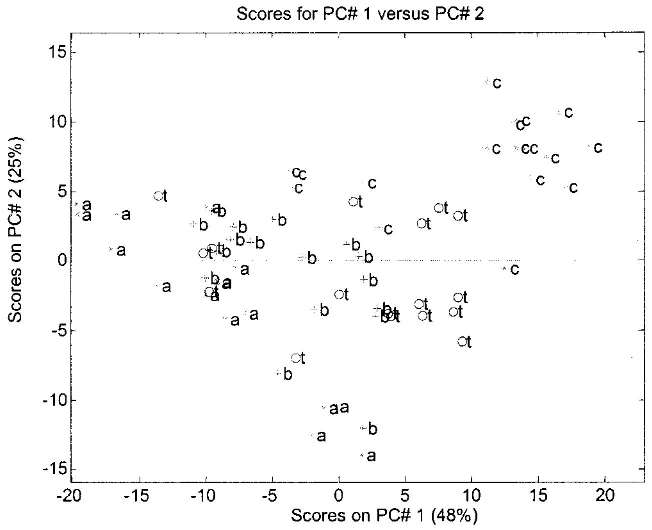


Fig. 10. PCA plot for test dataset: c = cedar; t: tobacco; c: coffe; a:air.

reference air for four additional heater periods. During all three cycles, data is collected at a pre-specified sampling frequency of 8 Hz, as defined in a setup file.

A. Classical Feature Extraction

As a first evaluation, feature extraction was done manually by using information from the last two periods of each cycle (reference, sampling and purge). The last five time samples (625 ms) for each temperature in one period were used as features, resulting in a 60-dimensional feature vector per sensor. The resulting PCA scatterplot is shown in Fig. 10. Large variance and class overlap can be observed. This is mainly caused by a lack of control in the sampling procedure, temperature conditions, and insufficient warm-up time. The goal of the signal processing stage will be to overcome this variability and calculate a stable model for classification.

The first components provide little discriminatory information, whereas later components with smaller eigenvalues (smaller variance) tend to provide better performance. This indicates that the principal variation in the data is introduced not by the target gas but by other dispersion sources, possibly changes in absolute concentration or differences in warm-up time. Better results could be obtained by using a supervised method such as LDA [28]. In this work we test the performance of a GMM operating on a PCA projection from a high dimensional space.

In our experiments, the best performance is achieved when projecting to six principal components, which correspond to 97% of the total variance, resulting in 94% successfully classified samples on test data. For comparison purposes, using a k-NN classifier we can only achieve 84%, as observed in Fig. 11. The confusion matrix for this test set is included in Table III. All the results are obtained using leave-one-out cross-validation. Good classification performance is observed even when using a less-than-careful sampling procedure, which is nonetheless representative of a data collection scenario in the field. This indicates that GMM is capable of accurately modeling the underlying multi-modal and multivariate distribution of the data.

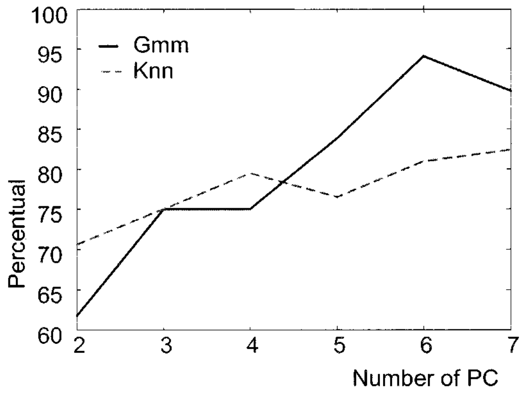


Fig. 11. Classifier performance with validation data (leave one out).

B. SFFS Feature Selection

To improve the performance of the system we decide to use SFFS as a feature selector. Let us consider as the initial feature set the 8 Hz samples of the last two periods of each cycle. Due to the large amount of data obtained (80 points per period = 480 features per sensor), subsampling is performed to obtain 12 equally spaced points per period. Sensor signals and the resulting features are shown for only one sensor in Fig. 12(a) (circled samples). The initial number of feature dimensions per sensor is 72 (3 cycles $\times$ 2 periods per cycle $\times$ 12 points per period), resulting in a total of 288 features for the entire sensor array (4 sensors $\times$ 72 points per sensor). Fig. 12 shows the original features after subsampling (as circles) for the second sensor in the array (SB-31-00).

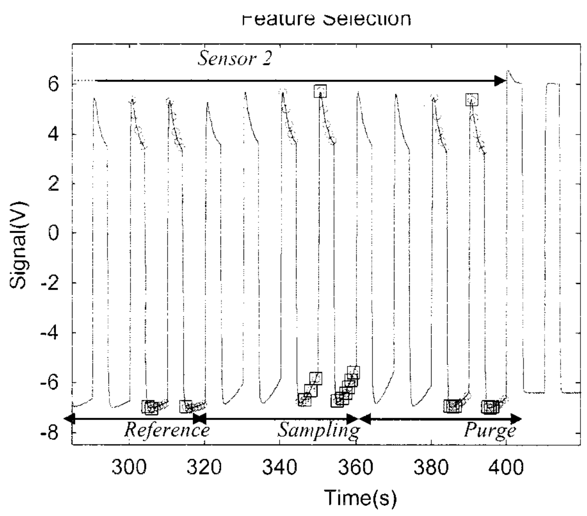
Taking in account all the 288 features in a sequential floating selection algorithm can be computationally very expensive. For this reason, the selection procedure is performed in two steps. The first step finds the features of each sensor that provide most classification related information. The second stage operates on the selections made by the first selection to find the final feature set.

The first step is done by applying sequential forward floating selection (SFFS) with a k-NN selection criteria and leave-one-out validation algorithm per each sensor separately. This is done in this way as parsing SFFS four times to a 72 length feature vector is much faster that parsing once over a 288 dimensional feature vector. A subset of the selected features per sensor is obtained, as shown in Fig. 12(a) by the square markers. Note that the majority of the selected points are gathered in the second cycle, which corresponds to sample "sniffing." Is also interesting that almost all selected points belong to the low temperature intervals. In Fig. 12(a), the square points represent this selection.

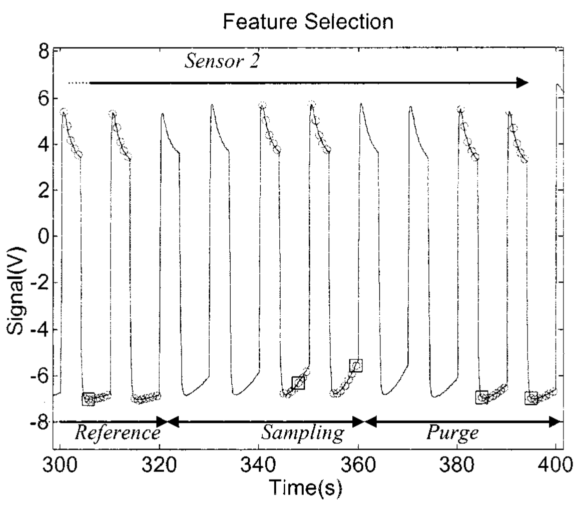
The second step is done by considering only those features that were selected on the preliminary stage, but considering the contribution of all the sensors as the new feature set. The final selection, shown in Fig. 12(b), comprises six features (out of 288), five of them from sensor 2, indicating that this sensor provides most of the discriminatory power. Note that the subset contains points from all three cycles. More precisely, two features originate from the actual "sniffing" of the target sample (one at each period), two from the last purging cycle (also one from each period) and one from the reference air cycle. The se-

TABLE III
CONFUSION MATRIX FOR TRAINING AND VALIDATION SETS IN GMM AND 3-NN. SIX PRINCIPAL COMPONENTS USED (VS: VALIDATION SET; TS: TRAINING SET)

	3-NN (VR=83.8%, TR=91.2%)				*GMM (VR=94%, TR=100%)*			
Predicted\Real	***Air***	***Cedar***	***Coffee***	***Tobacco***	***Air***	***Cedar***	***Coffee***	***Tobacco***
(VS)Air	14	2	0	2	15	2	0	0
(VS)Cedar	2	13	0	2	2	15	0	0
(VS)Coffee	0	0	17	0	0	0	17	0
(VS)Tobacco	1	2	0	13	0	0	0	17
(TS)Air	15	1	0	2	17	0	0	0
(TS)Cedar	3	16	0	1	0	17	0	0
(TS)Coffee	0	0	17	0	0	0	17	0
(TS)Tobacco	0	0	0	14	0	0	0	17



(a)



(b)

Fig. 12. (a) SFFS feature selection applied to sensor 2. Circles represent the decimated points, while the squares represent the union of the selected points after first pass through SFFS algorithm. (b) Final feature selection by SFFS algorithm (sensor 2). Circles represent the decimated points, while the squares represent the final selection of the points of the sensor 2.

lection of features from the reference and purge cycles suggests that this information is being used to remove common-mode components (e.g., drift, temperature) from the second cycle. Of the final six selected points, five belong to sensor 2, one to sensor 1 and none to the third and fourth sensors. Using this features and k-NN a classification rate of 98% is achieved using leave-one-out validation and no dimensionality reduction.

C. Landfill Site Data Set

The aim of this final section is to study the feasibility of remote bad odor detectors in landfill sites. For this purpose, we will apply a mixture model classifier against dataset provided by the Fondation Universitaire Luxembourgeoise (FUL).

The dataset was collected during the last week of July, 2000. All measurements were taken from 9 am to 6 pm. An assessment made by the operator, as well as data from CH4 and H2S analyzers were also provided. In addition, some meteorological conditions were measured, including wind speed, wind direction, rainfall, temperature and atmospheric pressure in a weather station. Electronic nose, gas analyzers, and weather station were located in the same shelter at the periphery of the landfill, 10 m far from the selective odor sources in the East direction. The target odors were biogas odor and waste odor.

TABLE IV
PERCENTAGE VARIANCE CAPTURED BY PCA. ALL DATA

Principal Component Number	*% Variance Captured*	*% Variance Captured Total*
1	60.59	60.59
2	24.30	84.88
3	11.02	95.90
4	2.34	98.24
5	1.21	99.45
6	0.30	99.75
7	0.22	99.97
8	0.03	100.00

The electronic nose used consists of six Figaro sensors placed in a compact metal enclosure (16 $\times$ 5 cm). The electronics provide temperature reading from two thermistors (NTC), one located inside the chamber and the other in the instrument case.

Both electronic nose and gas analyzers collect ambient air from 3.5-meter-high PFA tubing. The electronic nose cycle is as follows: reference air coming from a Tedlar bag is taken every 5 min and ambient air sampled during another 5 min at 150 ml/min. The Tedlar bags contained odorless synthetic air, filled in the laboratory. We build a feature dataset using the six gas sensor signals and the two temperatures. PCA shows that most

TABLE V
CONFUSION MATRIX FOR TRAINING AND VALIDATION SETS IN GMM AND 3-NN. FIVE PRINCIPAL COMPONENTS USED (VS: VALIDATION SET, TS: TRAINING SET)

	3-NN (VR=85.1%, TR=88.9%)			*GMM (VR=86.4%, TR=91.3%)*		
Predicted\Real	***Waste***	***Odourless***	***BioGas***	***Waste***	***Odourless***	***BioGas***
(VS)Waste	38	4	1	42	6	2
(VS)Odourless	6	21	0	3	19	0
(VS)BioGas	1	0	10	0	0	9
(TS)Waste	40	4	1	38	2	0
(TS)Odourless	3	21	0	6	23	0
(TS)BioGas	1	0	10	0	0	11

variance is contained in the first five components, as shown in Table IV. Data is mean centered, scaled to unit variance, and PCA is used as first step to coarsely reduce the dimensionality of sensor space down to five dimensions. The optimum number of components d is determined through leave-one-out cross validation using k-NN as classifier.

A significant dependence of GMM performance on the number of principal components is found. The addition of components actually degrades the performance of the classifier. This occurs when the number of samples used for training is small relative to the number of principal components used by PCA (number of features seen by the classifier). This is a normal phenomena related to the peaking phenomenon [29] and could be improved using discriminant analysis instead of principal component analysis as a dimensionality reduction step. The resulting confusion matrix for leave-one-out cross-validation is shown in Table V, where the top classification performance is found to be 86.4% on validation data. Training dataset showed a slightly overfitted 91.3%. For comparison purposes, using a k-NN classifier we obtained a classification rate of 85% for validation set.

Note that, although the results are similar, the resources required to calculate the EM loop, both in memory and computationally, are quite lower than for k-NN during recall. Using k-NN we are forced to have all the data in memory, whereas with GMM we hold only a model distribution of data. Computational needs during recall are approximately three times greater for k-nn than GMM (using test bench of 4 classes at 150 points/class at five dimensional space)

V. CONCLUSION

This work has shown a hardware and software solution for a portable and modular electronic nose system that takes full advantage of current embedded computing technologies. It is anticipated that these platforms will become widely popular in coming years as they provide a cost-effective solution for applying advanced signal processing techniques in standalone or portable electronic noses. An example of the software capabilities ***on-board*** the ipNose instrument have been shown, these include SFFS and Gaussian mixture model trained via EM algorithm. This is applied to feature selection by means of SFFS, feature extraction and classification of odors. Our results show that feature selection techniques such as SFFS can be extremely useful when applied to temperature modulated sensor arrays, as they are capable of to selecting not only the sensors but also the operating temperatures that provide most of the discriminatory information. A simple but effective classifier based on Gaussian mixture models has been and tested on experimental data from the ipNose and an environmental-monitoring dataset kindly donated by FUL, indicating that our GMM approach generalizes well to different e-nose platforms.

ACKNOWLEDGMENT

The authors want to thank specially to Dr. J. Nicolas (Fondation Universitaire Luxembourgeoise) for providing field test datasets.

REFERENCES

[1] J. Mitrovics, H. Ulmer, U. Weimar, and W. Göpel, "Modular sensor systems for gas sensing and odor monitoring: The MOSES concept," in *ACS Symp. Series: "Chemical Sensors and Interfacial Design*, vol. 31, 1998, pp. 307–315.

[2] J. W. Gardner and P. Bartlett, "Electronic Noses: Principles and Applications,", Oxford, 1999.

[3] [Online]. Available: http://www.pc104.org

[4] A. Romanenko and J. A. A. M. Castro, *Comput. Chem. Eng.*, vol. 24, no. 2–7, pp. 1063–1068, 2000.

[5] Y. Hiranaka, T. Abe, and H. Murata, "Gas-dependant response in the temperature transient of SnO_2 gas sensors," *Sens. Actuators B*, vol. 9, no. 3, pp. 177–177, 1992.

[6] S. Marco, A. Pardo, T. Sundic, A. Perera, and J. Samitier, "Opportunities for smart noses: What signal processing can do for you?," in *Proceedings EURODEUR*, Paris, Mar. 2001.

[7] S. Orfanidis, *Introduction to Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[8] S. Hirobayashi, H. Kimura, and T. Oyabu, "Dynamic model to estimate the dependence of gas sensor characteristics on temperature and humidity in environment," *Sens. Actuators B*, vol. 60, no. 1, pp. 78–82, 1999.

[9] S. Haiking, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[10] T. C. Pearce, "Computational parallels between the biological olfactory pathway and its analogue 'The electronic nose': Part II. Sensor-based machine olfaction," in *BioSyst.*, 1997, vol. 41, pp. 69–90.

[11] S. Marco, A. Pardo, A. Ortega, and J. Samitier, "Gas identification with tin oxide sensor array and self organizing maps: Adaptive correction of sensor drifts," *IEEE Trans. Instrum. Meas.*, vol. 47, pp. 316–320, Feb. 1998.

[12] T. Eklov, P. Martensson, and I. Lundstrom, "Selection of variables for interpreting multivariate gas sensor data," *Anal. Chim. Acta 381*, pp. 221–232, 1999.

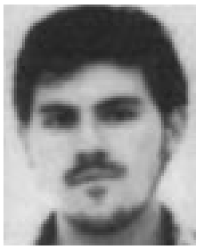
[13] A. Jain and D. Zongker, "Feature selection: Evaluation, application and small sample performance," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 153–158, Feb. 1997.

[14] P. Somol, P. Pudil, J. Novovicová, and P. Paclík, "Adaptive floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 20, pp. 1157–1163, Nov. 1999.

[15] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain, "Dimensionality reduction using genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 4, pp. 164–171, July 2000.

[16] A. Pardo, S. Marco, A. Ortega, A. Perera, T.Šundic, and J. Samitier, "Methods for sensors selection in pattern regognition," *Proc. ISOEN*, pp. 83–88, 2000.

[17] D. M. Titterington, A. F. M. Smith, and U. E. Makov, *Statistical Analysis of Finite Mixture Distributions*. New York: Wiley, 1985.
[18] G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*. New York: Marcel Dekker, 1988.
[19] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via EM algorithm," *J. Roy. Statist. Soc. B*, vol. 39, no. 1, pp. 1–38, 1977.
[20] S. Richardson and P. Green, "On Bayesian analysis of mixtures with unknown number of comnponents," *J. Roy. Statist. Soc. B*, vol. 59, pp. 731–792, 1997.
[21] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1999.
[22] M. A. T. Figuereido and A. K. Jain, "Unsupervised selection and estimation of finite mixture models," *Int. Conf. Pattern Recognit.*, vol. 2, pp. 87–90, Sept. 2000.
[23] C. Wu, "On the convergence propierties of the EM algorithm," *J. Roy. Statist. B.*, vol. 45, no. 1, pp. 47–50, 1983.
[24] N. Friedman and M. Goldszmidt, "Building classifiers using Bayesian networks," in *Proc. Nat. Conf. Artificial Intell. (AAAI)*, 1996, pp. 1277–1284.
[25] W. F. Massy, "Principal component regression in exploratory statistical research," *J. Amer. Statist. Assoc.*, vol. 60, pp. 234–246, 1965.
[26] H. Wold, "Soft modeling by latent variables: The nonlinear iterative partial least squares approach," in *Perspectives in Probability and Statistics*, J. Gani, Ed. London, U.K.: Academic, 1975.
[27] T. Sundic, S. Marco, A. Perera, A. Pardo, S. Hann, N. Bârsan, and U. Weimar, "Fuzzy inference system for sensor array calibration: Prediction of CO and CH4 levels in variable humidity conditions," *Chemometr. Intell. Lab. Syst.*, submitted for publication.
[28] A. Perera, R. Gutierrez-Osuna, and S. Marco, "IpNose: A portable electronic nose nose based on embedded technology for intensive computation and time dependent signal processing," in *Int. Symp. Electron. Noses (ISOEN2001) abs. 1082*.
[29] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Handbook of Statistics.* Amsterdam, The Netherlands, 1987, vol. 2, pp. 835–855.



Alexandre Perera received the degree in physics in 1996 and the degree in electrical engineering in 2001 from the University of Barcelona, Barcelona, Spain, where he is currently pursuing the Ph.D. degree.

His main research interests include signal processing and pattern recognition techniques applied to electronic noses. He is currently working in the use of embedded systems as a platform for autonomous advanced signal processing for machine olfaction.

Teodor Sundic received the diploma in electrical engineering from the University of Belgrade, Belgrade, Yugoslavia, in 1996, and is currently pursuing the Ph.D. degree at the University of Barcelona, Barcelona, Spain.

His research interests include pattern recognition techniques, fuzzy systems, artificial neural networks, feature selection and extraction algorithms applied to electronic noses, and gas detection devices.

Antonio Pardo received the degree in physics in 1991 and the Ph.D. degree in 2000 from the University of Barcelona, Barcelona, Spain.

During his Ph.D. studies, he worked in system identification with applications in gas sensor systems. His research interest focused on signal processing for gas sensors and pattern recognition.

Ricardo Gutierrez-Osuna (M'00) received the B.S. degree in industrial/electronics engineering from the Polytechnic University of Madrid, Spain, in 1992, and the M.S. and Ph.D. degrees in computer engineering from North Carolina State University, Raleigh, in 1995 and 1998, respectively.

From 1998 to 2002, he served on the faculty at Wright State University, Dayton, OH. He is currently an Assistant Professor with the Department of Computer Science at Texas A&M University. His research interests include pattern recognition, machine learning, biological cybernetics, machine olfaction, speech-driven facial animation, computer vision, and mobile robotics.

Santiago Marco received the degree in Physics from the Universitat de Barcelona, Barcelona, Spain, in 1988. In 1993, he received the Ph.D. (honor award) degree from the Departament de Física Aplicada i Electrònica, Universitat de Barcelona, for the development of a novel silicon sensor for in-vivo measurements of the blood pressure.

He has been an Associate Professor with the Departament d'Electronica, Universitat de Barcelona, since 1995. From 1989 to 1990, he was worked with the electro-optical characterization of deep levels in GaAs. From 1990 to 1993, he was a regular visitor of the Centro Nacional de Microelectrònica, Bellaterra, Spain. In 1994, he was Visiting Professor with the Universita di Roma, Tor Vergata, working with data processing for artificial olfaction. He has published about 40 papers in scientific journals and books, as well as more than 80 conference papers. His current research interests are twofold: chemical instrumentation based on intelligent signal processing and microsystem modeling by FEM and HDLA languages.